



**TECHNICAL
DOCUMENTATION**

for

U N I C O D E

**Automatic Programming System for
Univac Scientific 1103A and 1105**

Volume III

April , 1961

PX 1790

Remington Rand Univac[®]

DIVISION OF SPERRY RAND CORPORATION

UNIVAC PARK, ST. PAUL 16, MINNESOTA

VOLUME I

	Page
Table of Contents	I-v
I. INTRODUCTION.	3
II. GENERAL	
1. UNICODE Service Routines.	7
2. Library Routines.	49
3. UNICODE System Tape Package	123
4. UNICODE Sample Coding	153
5. UNICODE Card Input.	163
6. Statistical Miscellany.	185
III. TRANSLATION AND CORRECTION	
1. UNICODE Sentinel Blocks	203
2. Tape Merge.	217
3. Translation Phase	
a. Translation Subroutines	291
b. Translators	434

VOLUME II

Table of ContentsII-v
III. TRANSLATION AND CORRECTION	
3. Translation Phase	
b. Translators (cont.)	569
IV. GENERATION PHASE	
1. Generation Set-up and Drum Loader	949
2. Generation Subroutines.	959
3. Generators.	1013

VOLUME III

Table of Contents	III-v
IV. GENERATION PHASE	
3. Generators (cont.)	1193
V. ALLOCATION PHASE	
1. Segmentor	1461
2. Allocator	1551
3. Initialization Generator.	1607
VI. PROCESSING PHASE.	1671
VII. PROGRAM LISTING PHASE	1747

VOLUME III

TABLE OF CONTENTS

IV. GENERATION PHASE

3. GENERATORS (cont.)

EQUATION Generation

EQUATION Generation No. 1

Write-Up	1193
Flow Charts	1203
Coding	1211

EQUATION Generation No. 2

Write-Up (Also for EQUATION Generation No. 3).	1230
Flow Charts	1234
Coding	1297

EQUATION Generation No. 3

Flow Charts	1352
Coding	1403

V. ALLOCATION PHASE

1. SEGMENTOR

a. Segmentation Setup	1461
b. Segmentation	
Write-Up	1464
Flow Charts	
Phase I	1467
Phase II	1477
Coding	
Phase I	1481
Phase II	1502

2. ALLOCATOR

a. Allocation Setup	
Write-Up	1551
Flow Charts	1552
Coding	1554
b. Allocation Phase	
Write-Up	1561
Flow Charts	1566
Coding	1582

3. INITIALIZATION GENERATOR

Initialization Generation Setup 1607

Initialization

Notes 1612

Generation

Flow Charts 1619

Coding 1622

Section I

Flow Charts 1629

Coding 1639

Section II

Flow Charts 1655

Coding 1656

Control Section for Object Program

Write-Up 1659

Flow Charts 1662

Coding 1664

VI. PROCESSING PHASE

Notes 1671

Processor Setup Coding 1677

Flow Charts 1678

Coding 1705

VII. PROGRAM LISTING PHASE

Notes 1747

Program Listing Setup Coding 1759

Flow Charts 1761

Coding 1796

EQUATION GENERATION

Equation Generation No. 1

The coding for an equation is generated in three stages numbered 1, 2 and 3. Number 1 produces a sorted list of symbols, No. 2 eliminates some redundant calculations, and No. 3 produces the coding.

The idea of No. 1 is to add parentheses to the equation (which has been "strung out" one call word per computer word by the equation translator) and number call words by use of the parentheses in the expression. The numbered call words are then sorted and generator No. 2 takes over.

Thus there are three passes made by No. 1: processing (adding parentheses), numbering symbols, and sorting. An explanation of each of these follows a description of the lists.

The six lists made up or used by this routine are as follows:

1) Translation List (WL)

This is the input to the routine and is produced by the equation translator. It contains one call word per computer word, the call words being in the v addresses, except that an open parenthesis is a 1 in the u address and a closed parenthesis is a 2 in the u address. See the equation translation description for a more detailed explanation.

2) Processed List (PR)

The WL list is examined one call word at a time and parentheses are added where needed to produce this list.

3) Numbered List (WL (same region as Translation List))

The Processed List entries are picked up one at a time, starting with the last symbol in the list, numbered, and then transferred to the Numbered List, with the exception of open and closed parentheses which are used to alter the Numbers of Symbols (NS) List and are not sent to the Numbered List. (See descriptions of numbering and Numbers of Symbols List.)

4) Sorted List (PR (same region as Processed List))

This is the list produced by sorting the Numbered List so that larger numbers are at the beginning of the list. It is the output of the routine.

5) Parentheses List (PL)

This is a two-word-per-item list which contains a code for the type of open parenthesis in the operation portion of the first word and the level bit in one of the remaining 30 bits. The second word contains the Processed List address of the parenthesis in the u address of the word. This list contains only items for open parentheses.

Op	u	v
0 X	(level bit)	
0 0	(P)	

X = type of parenthesis

- X = 0 - "not special"
- X = 1 - level
- X = 2 - term
- X = 3 - Library
- X = 4 - POW

P = address of parenthesis in PR list.

6) Numbers of Symbols List (NS)

This list is used when producing the Numbered List. In the Processed List every parenthesis will have a count in the v address to indicate how many parentheses are at this point. For example the following words might appear in the Processed List (not consecutively):

Op	u	v	
0 0	0 0 0 0 1	0 0 0 0 6	Six open parentheses
0 0	0 0 0 0 2	0 0 0 0 4	Four closed parentheses

For every closed parenthesis encountered in the Processed List, numbers are added to the NS List. The number of numbers added is equal to the count in the v address of the closed-parenthesis word. Open parentheses are handled similarly except that numbers are deleted from the NS List. The numbers in the NS List are in the u addresses of the words. For example, at one time the NS List may look as follows:

	Op	u	v
NS	0	1	0
NS 1	0	2	0
NS 2	0	3	0
NS 3	0	4	0
NS 4	0	10	0
NS 5	0	11	0
NS 6	0	16	0
NS 7	0	24	0

The last number in the list (24 in this one) is always the number added to a symbol call word to make up the numbered symbol for the Numbered List. (The length of list NS varies, of course.) The last number in the list is always the largest still in the list but there may have been larger numbers previously. Parentheses are never put in the Numbered List; they are merely used to alter the NS List. Suppose we now encounter an open parenthesis with a count of 5. Five is subtracted from the last address (NS7) and the last address now becomes NS2 and the number to assign symbols is 3. Later we encounter a closed parenthesis with a count of 7. Numbers are added to the list starting with 25 since we have already used 1 to 24. Since we must add 7 numbers the list becomes:

	Op	u	v
NS	0	1	0
1	0	2	0
2	0	3	0
3	0	25	0
4	0	26	0
5	0	27	0
6	0	30	0
7	0	31	0
10	0	32	0
NS11	0	33	0

and the next symbol (if not a parenthesis) will be numbered 33.

The explanation of the three passes follows:

Processing:

A level bit is kept up to date at all times. It starts at the rightmost bit position and is shifted left by one every time an open parenthesis or open absolute-value sign is encountered, and right by one for every closed paren-

thesis or absolute-value sign. One may write up to 29 open parentheses and/or absolute-value signs before he must close some. That is, he may write symbols on the 29th level but not on higher levels.

There are five types of open parentheses added to the Processed List. These are the level, term, library, POW, and anticipation (for want of a better name). A level and a term parenthesis are added to the Processed List every time an open parenthesis, open absolute-value sign or comma is encountered in the Translation List. A level parenthesis is put in the Processed List before the first symbol is picked up from the Translation List and, when the equals sign is encountered, level and term parentheses are also added. A term parenthesis is added at the beginning of each term, i.e., after a binary + or - sign.

A library (LIB) parenthesis is added before each Library Routine symbol unless there is already an unclosed library parenthesis (on the same level) in the list.

When POW is encountered, the last open parenthesis is changed to a POW parenthesis in the Parenthesis List (PL).

The anticipation parenthesis is added in the following places:

- 1) After every multiplication, division or unary minus sign in anticipation of the next operation being POW. (If it isn't POW, the anticipation parenthesis will not alter the interpretation.)
- 2) Before and after a library call word when there is already a library parenthesis on this level. This is to handle the case:

$$\begin{array}{c} (\text{LIB} (\text{LIB} (X))) \\ \uparrow \quad \quad \quad \uparrow \quad \quad \quad \uparrow \\ \text{Library} \quad \quad \quad \text{A} \quad \quad \quad \text{A} \end{array}$$

where all of the parentheses have been added, i.e., none were originally written in the expression. This puts the rightmost Library Routine on the highest level.

- 3) After a library call word so the operands will be assigned larger numbers than the library call word.
- 4) Before every unary minus to associate the unary minus with the operand which follows.

The preceding discussion deals with open parentheses. When closing parentheses, a closed parenthesis with a count of zero is added to the Processed List and open parentheses in the Parentheses List are examined one at a time starting with the last parenthesis item in the list. Parentheses are closed by adding one to the count of both the closed and open parentheses in the Processed List. If the parenthesis just closed is not of the type sought, it is deleted from the Parentheses List by subtracting 2 from its address in the Parentheses List. This puts the next parenthesis "on deck" and the process continues until the type of parenthesis sought is closed. After this the parenthesis is left "on deck" or deleted from the Parenthesis List depending on circumstances.

Following is a summary of what is done upon encountering each of the symbols of an equation in the Translation List ("level" means the level due to parentheses or absolute value signs written in the UNICODE Program.)

- Subscripted Variable - Anticipation parenthesis to Processed and Parentheses lists. Variable call word to Processed List.
- Library Routine - 1) Previous library parenthesis on same level, still in Parenthesis List: Anticipation parenthesis to lists. Library call word to Processed List. Anticipation parenthesis to lists.
- 2) No previous library parenthesis on same level, still in Parentheses List: Library parenthesis to lists. Library call word to Processed List. Anticipation parenthesis to list.
- P O W - 1) Previous POW parenthesis on same level, still in Parenthesis List: Close parentheses to POW parenthesis (leave POW parenthesis "on deck"). POW to Processed List.
- 2) Previous library parenthesis on same level, still in Parenthesis List: Close to library parenthesis and change it to a POW parenthesis in the Parenthesis List (leave POW parenthesis "on deck"). POW to Processed List.

- 3) No previous library or POW parenthesis. Close to last open parenthesis and change it to a POW parenthesis. POW to Processed List.
- Special powers (Square, Square Root, etc.) - Same as POW then: Close to POW parenthesis (leave "on deck").
- Open parenthesis and Open absolute value sign - Increase level. Level and term parentheses to lists. (Note that no open absolute value sign is put in Processed List.)
- Closed Parenthesis - Close to level parenthesis and delete it from Parenthesis List. Decrease level.
- Closed Absolute Value Sign - Close to level. Absolute value sign to Processed List. Close to level and delete from Parenthesis List. Decrease level.
- + or - sign - Close to level. + or - to Processed List. Term parenthesis to lists.
- Unary plus - Ignore.
- Unary minus - Anticipation parenthesis to lists. Unary minus to Processed List. Anticipation parenthesis to lists.
- Comma - Close to level parenthesis. Add level and term parenthesis to lists. (Note no comma is sent to Processed List.)
- Equals sign - Close to level parenthesis. Add level and term parentheses to lists. (Note no equals sign is sent to Processed List.)
- * or / sign - Close to term parenthesis. * or / to Processed List. Anticipation parenthesis to lists.
- Space period - Close to level parenthesis. Space period to Processed List. Jump to numbering routine.

In addition, indicator bits are kept for each term of the expression so ambiguous sequences can be recognized and a warning printed on the typewriter.

Then, if the programmer is not sure of the interpretation of UNICODE he can rewrite the sentence and put parentheses in the expression so he will be sure to get the correct interpretation. The following ambiguous terms are recognized (the interpretation of UNICODE is on the right):

A POW B POW C = (A POW B) POW C
A/B/C = (A/B) / C
LIB A POW B = (LIB A) POW B
LIB A*B = (LIB A) * B
LIB A/B = (LIB A) / B

Compilation continues after the warning is printed.

Numbering:

Call words are numbered by use of the last number in the Numbers of Symbols List (NS). The numbers in this list are in the u addresses, one number per word. Two things must be known to use this list:

1. The address of the last number in the list.
2. The largest number put in the list so far. (The last number in the list is the largest in the list but not necessarily the largest number which has been in the list for this equation.)

Once a number has been in the list and has been taken out, it will not appear in the list again. The first number put in the list is 1.

Call words and parentheses are picked up from the Processed List starting with the last call word (space period). Call words other than parentheses are numbered with the last number in the NS List; then the numbered call word is sent to the Numbered List.

When a closed parenthesis is encountered, numbers are added to the NS List, the number of numbers added being equal to the count associated with the closed parenthesis. Numbers which are added are equal to the largest number which is or has been in the list plus 1. The address of the last number in the list is increased by one for each number added to the list, of course.

When an open parenthesis is encountered, the count is subtracted from the address of the last number in the list, hence essentially deleting numbers from the list.

The space period is numbered zero.

Sorting:

The Numbered List is sorted, largest first, to produce the Sorted List which is the output of equation generator No. 1.

For example, consider the following equation as input to the routine.

$$F(I,J) = -X \text{ POW } Y + (\text{SIN } |u - v|) * W\Delta.$$

The Processed List would be as follows (numbers above parentheses are counts and letters below are types, where L = level, T = term, A = anticipation, B = library, P = POW.):

```
11 11 2 11 22 211 2 1      4 2 111 1 31 2 1 2 141 1 3
((F ((I) ((J)) (((-X) POW Y) + ( ((SIN ( ((u)-(v) | ))) * (w)Δ.
LA LT  LT  LTA A
                P      T LTB  A LT  T      A
```

Numbering the symbols:

Δ. is numbered zero and sent to Numbered List.

Symbol	NS List	Numbered List	
		Number	Symbol
Δ.		0	Δ.
3	1,2,3		
W		3	W
(¹	1,2		
*		2	*
)	1,2,4		
4	1,2,4,5,6,7,8		
)	1,2,4,5,6,7,8,9		
		9	
) ²	1,2,4,5,6,7,8,9,10,11		
V		11	V
(¹	1,2,4,5,6,7,8,9,10		
-		10	-
) ²	1,2,4,5,6,7,8,9,10,12,13		
U		13	U
(¹	1,2,4,5,6,7,8,9,10,12		
(³	1,2,4,5,6,7,8		
(¹	1,2,4,5,6,7		
SIN		7	SIN
(¹	1,2,4,5,6		
(¹	1,2,4,5		
(¹	1,2,4		
(²	1		
+		1	+
) ⁴	1,14,15,16,17		
Y		17	Y
POW		17	POW
)	1,14,15,16,17,18		

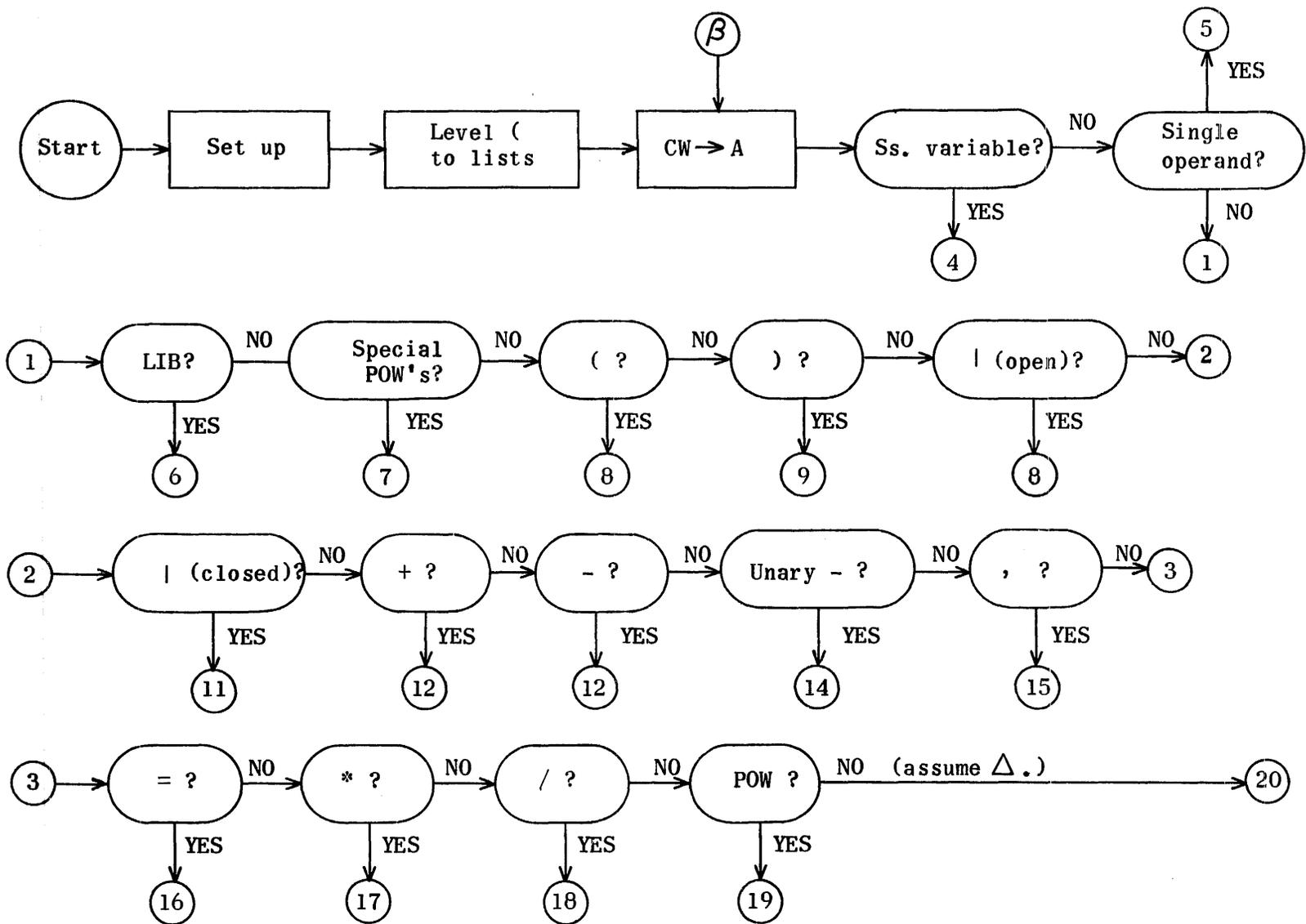
Symbol	NS List	Numbered List	
		Number	Symbol
X ²	1,14,15,16	18	X
- ¹	1,14,15	16	-
(¹	1,14		
(²	list empty		
) ²	19,20		
) ²	19,20,21,22		
J ¹	19,20,21	22	J
(¹	19,20		
) ²	19,20,23,24		
I ¹	19,20,23	24	I
(¹	19,20		
F ¹	19	20	F
(¹	list empty		

Note: Numbers over parentheses denote count of parentheses occurring at this point.

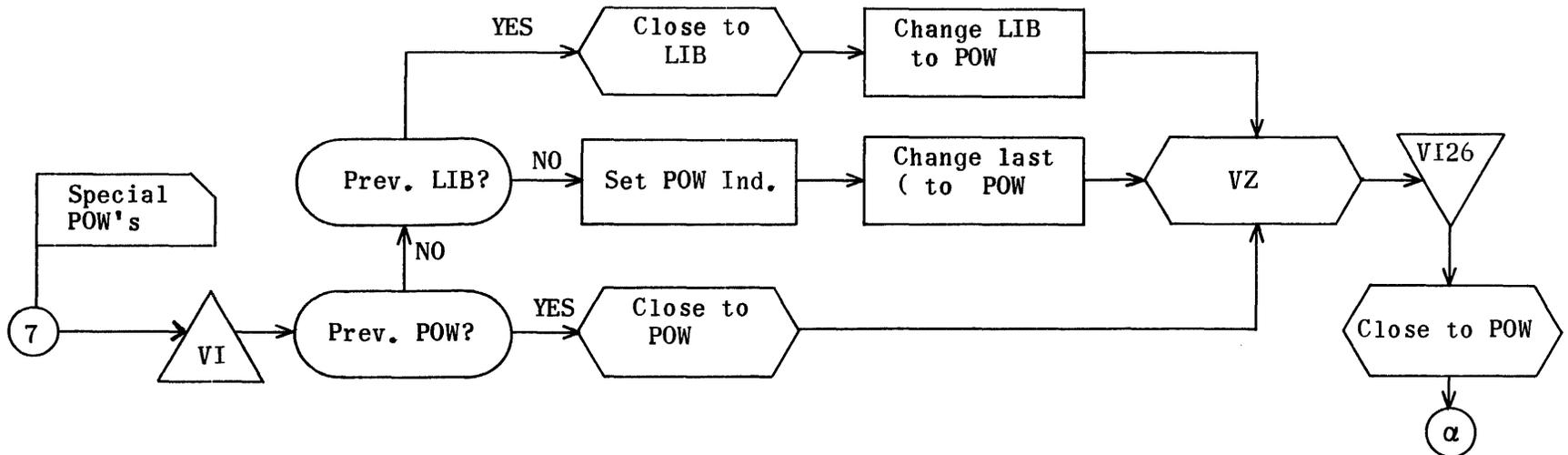
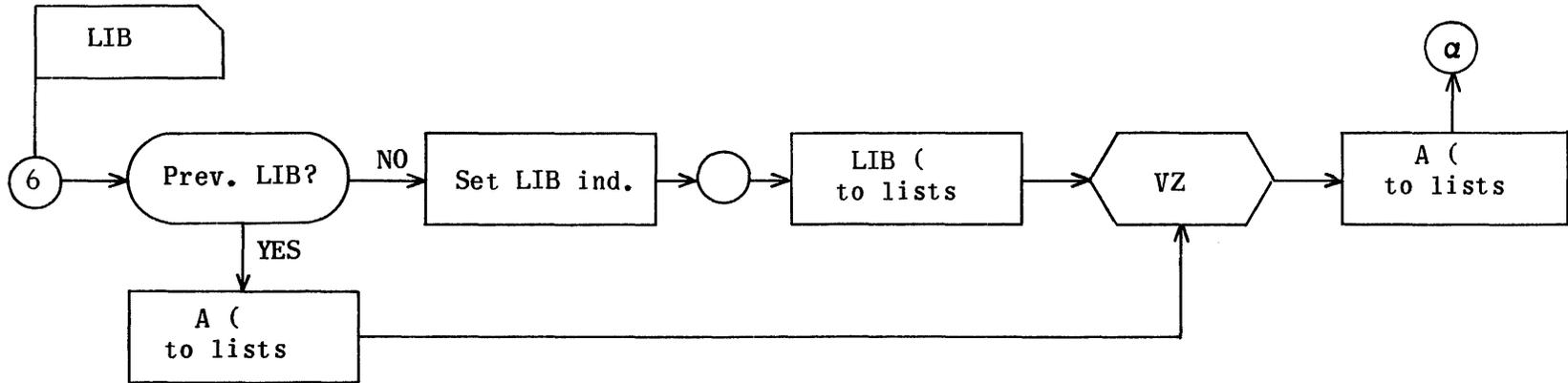
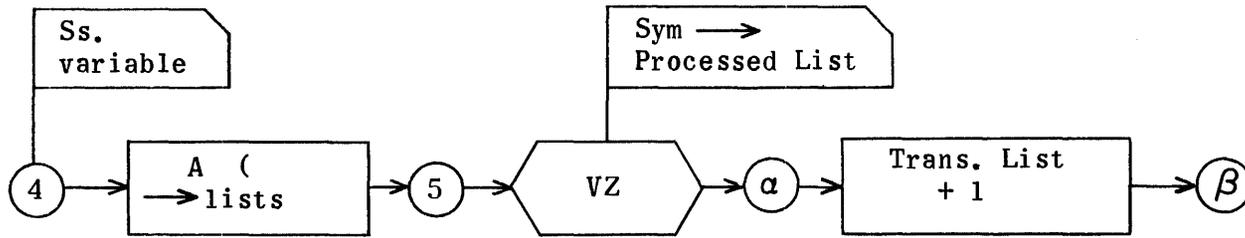
Sorted List:

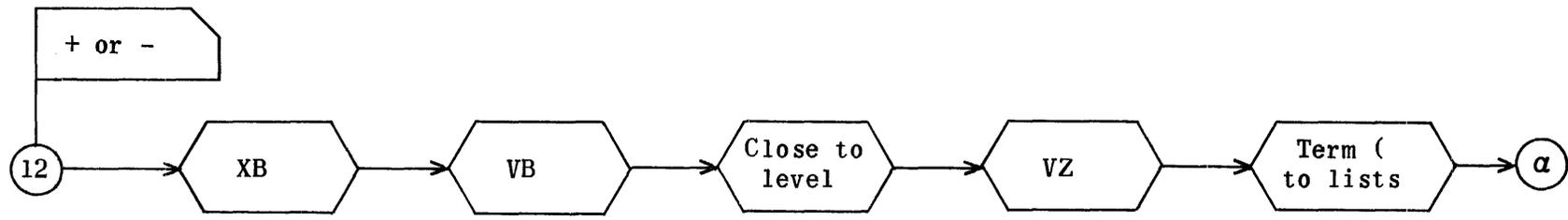
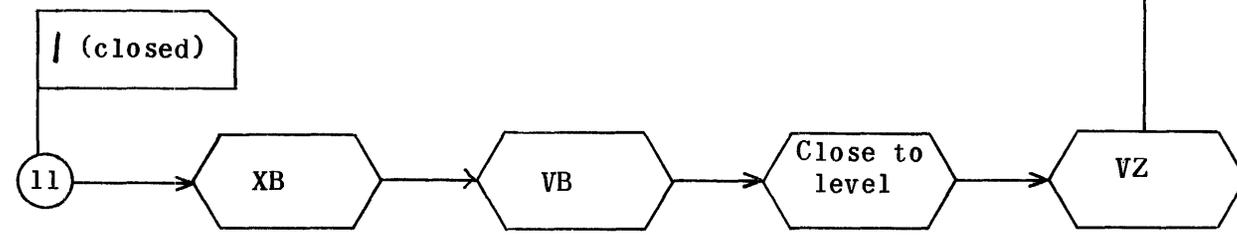
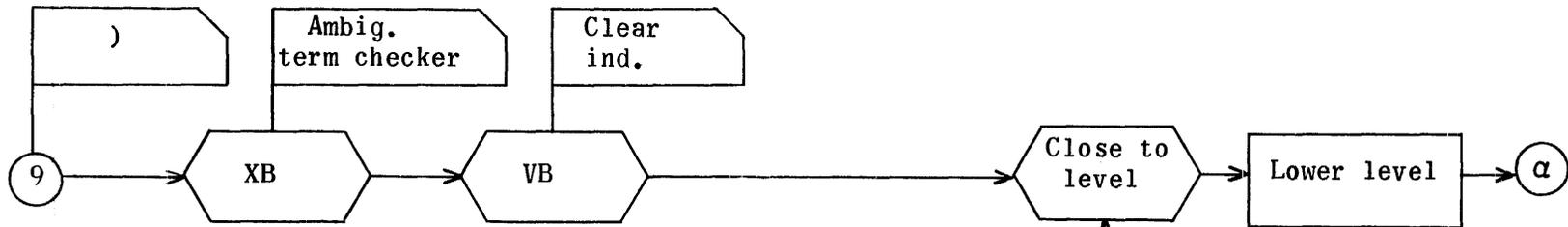
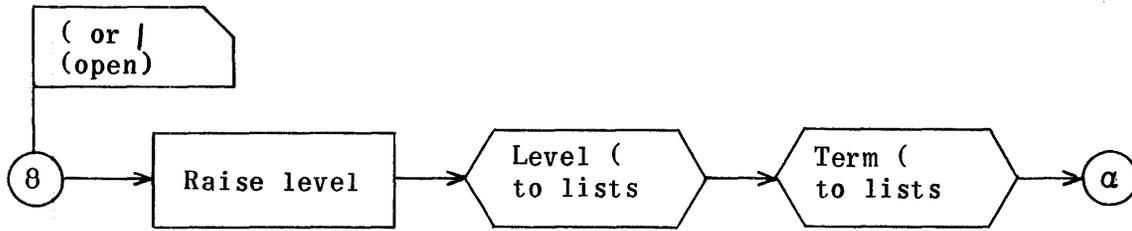
I	
J	
F	
X	
Y	} both numbered 17 but operands always have larger call words than operations.
POW	
-	Unary
U	
V	
-	Binary
SIN	
W	
*	
+	
Δ.	

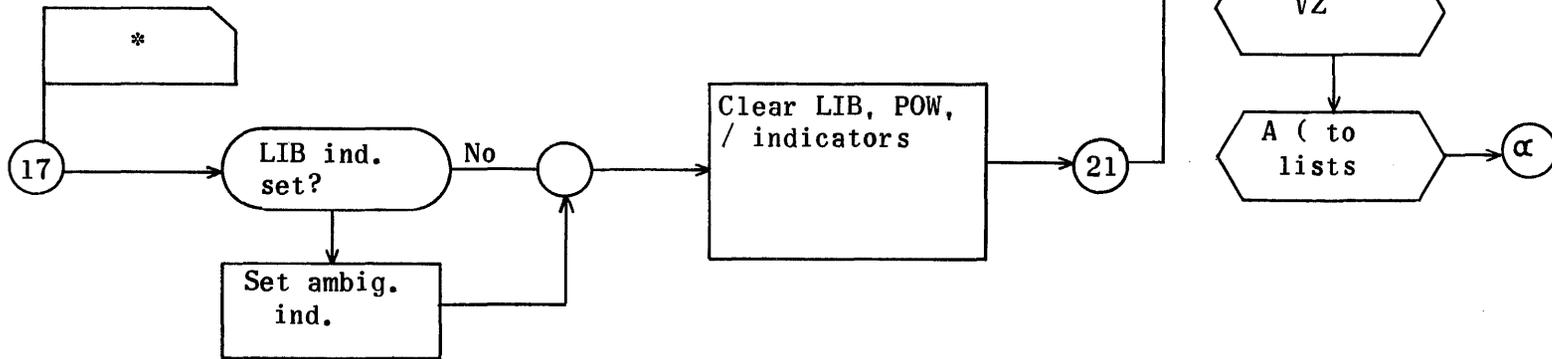
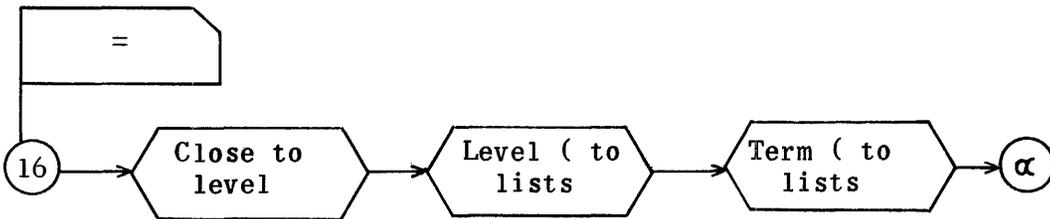
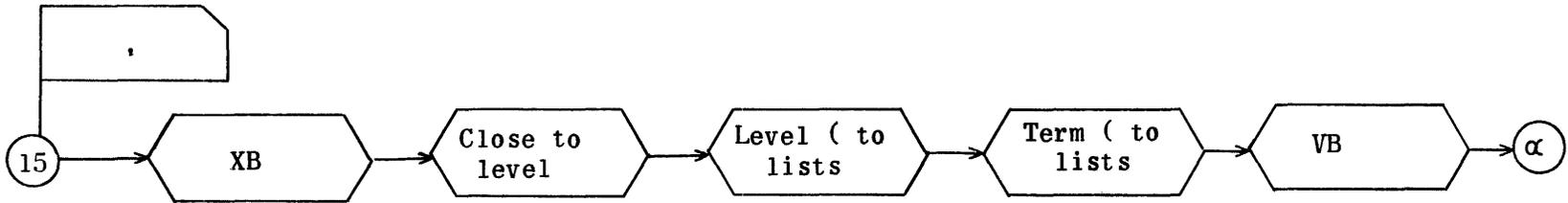
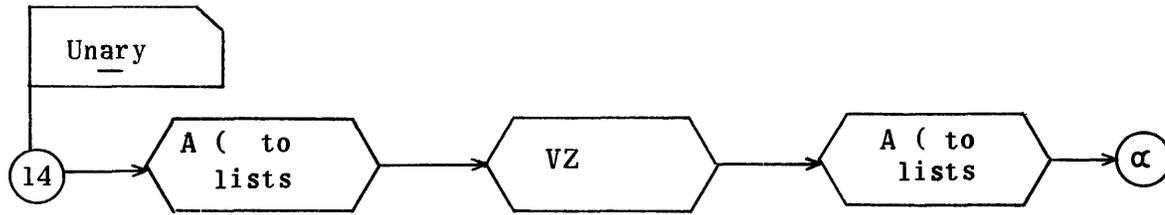
Equation Generation No. 1

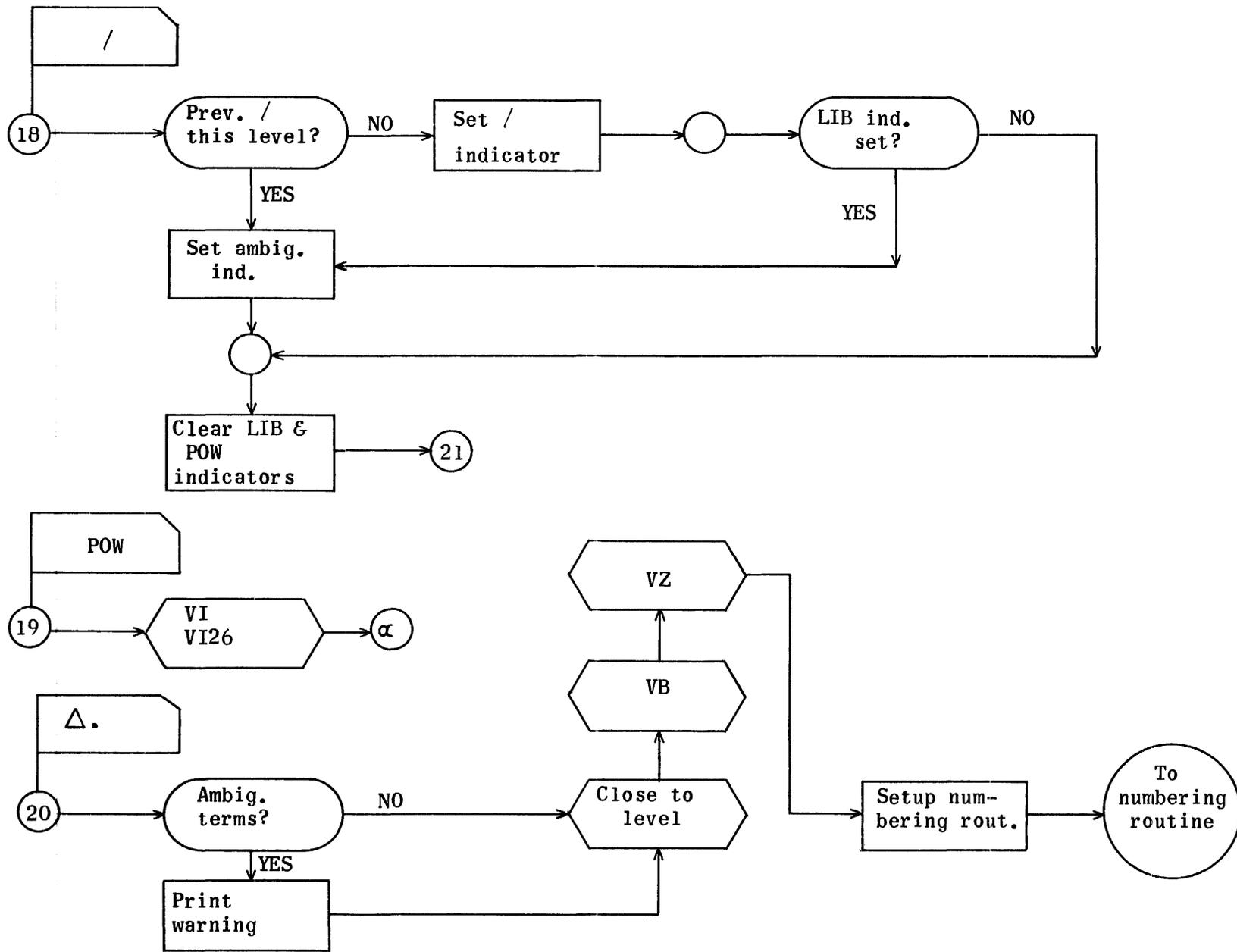


A = Anticipation

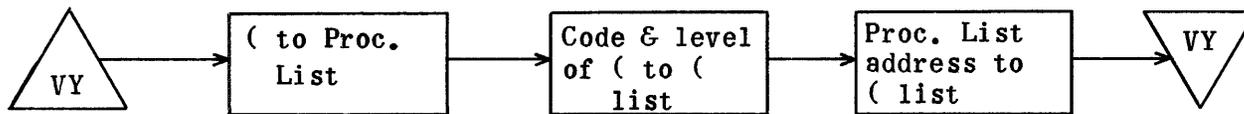




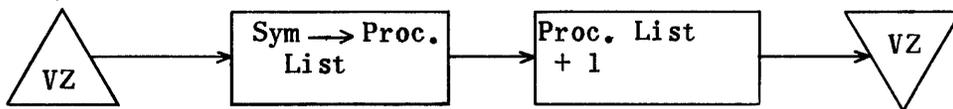




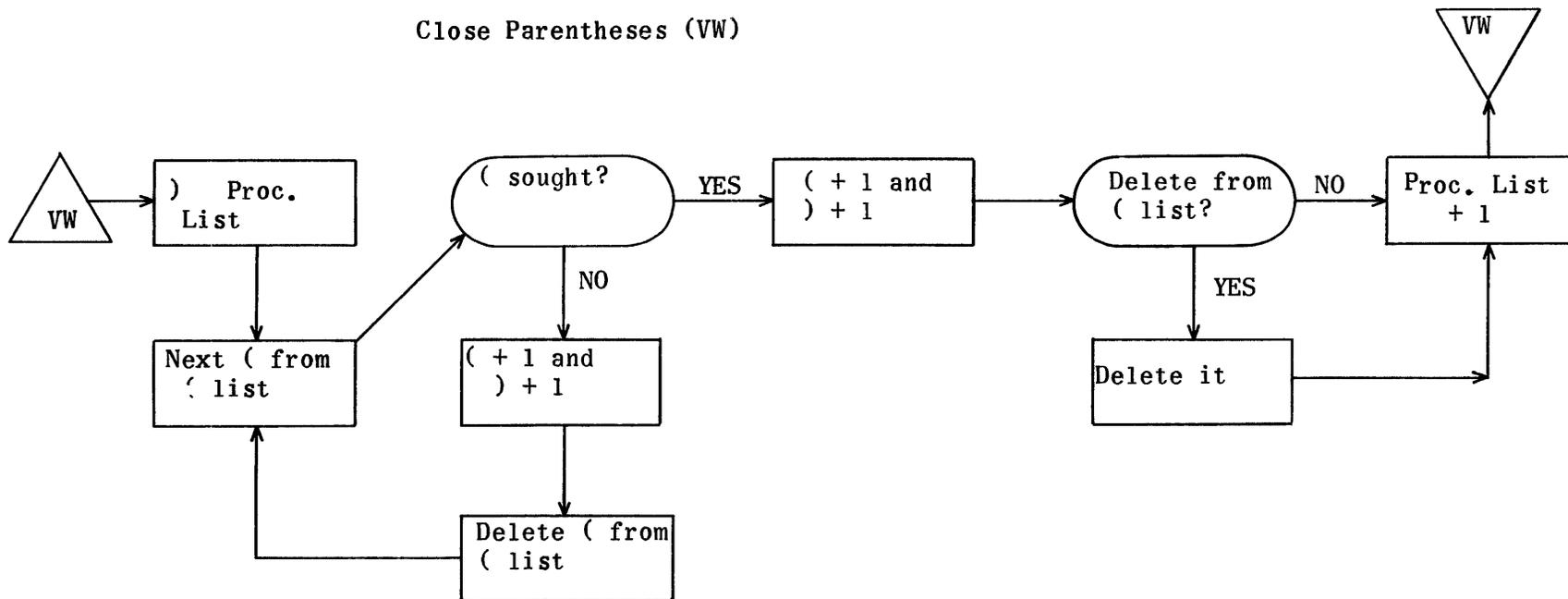
Open Parenthesis to Lists (VY)



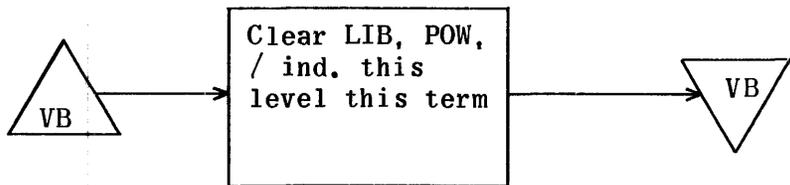
Symbol to Processed List (VZ)



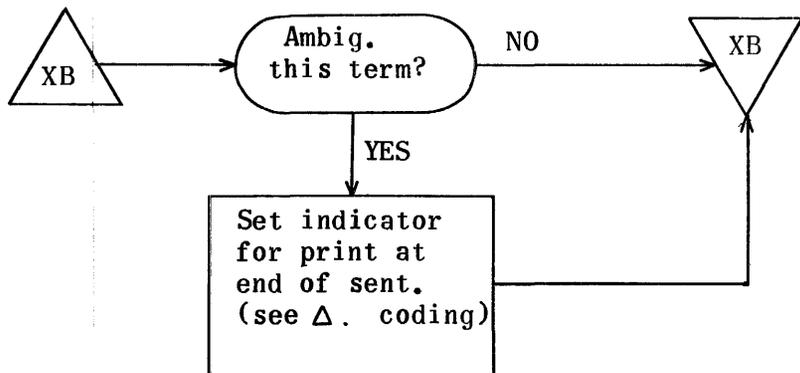
Close Parentheses (VW)



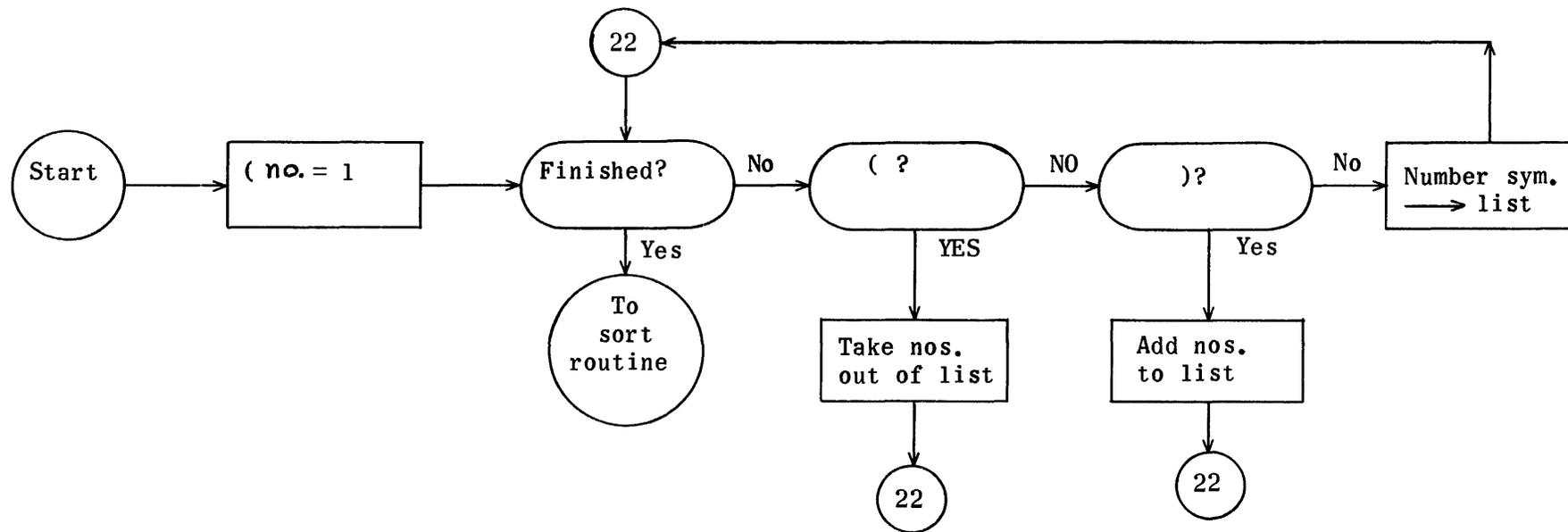
Clear Indicators (VB)



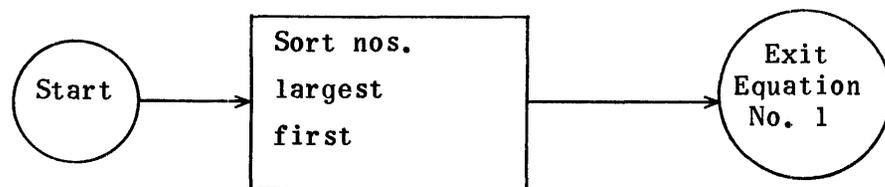
Check for Ambiguity (XB)



NUMBERING ROUTINE



Sort Routine



Equation Generator No. 1 Regions and Coding

Region	Address	Name or Symbol Handled
VD	2512	Setup
VB	2523	Clear Indicators
VC	2532	Constants
VE	2616	Switch
VF	2654	Subscripted Variable
VH	2660	Library Routine
VI	2674	Special POWS
VJ	2752	Open Parenthesis and Open absolute
VK	2760	Closed Parenthesis
VL	2770	Closed Absolute Value
VM	3003	+ or -
VN	3013	Unary -
VO	3020	Comma
VP	3033	=
VQ	3044	*
VR	3062	/
VS	3075	POW
VT	3077	Δ .
VU	3114	Numbering Routine
VW	3147	Close Parentheses
VX	3172	(+1 and) +1
VY	3202	Add Parenthesis to List
VZ	3220	Symbol to Processed List
XA	3226	Trans. List +1
XB	3230	Check for Ambiguity
XC	3236	Constants
SR	3244	Sort Routine
VA	3324	Variable
PR	3351	Processed List
NS	4351	Numbers of Symbols List
PL	5351	Parentheses List
WL	2242	Translation List
NT	2774	Close to level and Sym \rightarrow Processed List

Setup Equation Generation

	IA	VD			
0	MJ	0	(30000)	Exit	
1	RP	13025	VD3	}	Clear variables
2	TP	VC	VA		
3	TP	VC4	VA		Level bit
4	RP	30004	VD6	}	Set list addresses
5	TP	VC60	VA6		
6	TP	VC44	VY2	}	Add level (to lists
7	RJ	VY	VY1		
10	MJ	0	VE		→ (B)
	CA	VD11			

	IA	XA			
(a)	0	RA	VA6	VC1	Sym/wd list +1
	1	MJ	0	VE	→ (B)
		CA	XA2		

Translation Switch

(β)	<table border="0"> <tr><td>0</td><td>IA</td><td>VE</td><td></td></tr> <tr><td>1</td><td>TU</td><td>VA6</td><td>VE1</td></tr> <tr><td>2</td><td>QT</td><td>VC13</td><td>A</td></tr> <tr><td>3</td><td>EJ</td><td>VC13</td><td>VF</td></tr> <tr><td>4</td><td>EJ</td><td>VC14</td><td>VF2</td></tr> <tr><td>5</td><td>EJ</td><td>VC15</td><td>VH</td></tr> <tr><td>6</td><td>EJ</td><td>VC17</td><td>VI</td></tr> <tr><td>7</td><td>QT</td><td>VC7</td><td>A</td></tr> <tr><td>10</td><td>EJ</td><td>VC42</td><td>VJ</td></tr> <tr><td>11</td><td>EJ</td><td>VC43</td><td>VK</td></tr> <tr><td>12</td><td>TP</td><td>Q</td><td>A</td></tr> <tr><td>13</td><td>EJ</td><td>VC21</td><td>VJ</td></tr> <tr><td>14</td><td>EJ</td><td>VC22</td><td>VL</td></tr> <tr><td>15</td><td>EJ</td><td>VC40</td><td>VL</td></tr> <tr><td>16</td><td>EJ</td><td>VC23</td><td>VM</td></tr> <tr><td>17</td><td>EJ</td><td>VC24</td><td>VM</td></tr> <tr><td>20</td><td>EJ</td><td>VC25</td><td>VM</td></tr> <tr><td>21</td><td>EJ</td><td>VC26</td><td>VM</td></tr> <tr><td>22</td><td>EJ</td><td>VC27</td><td>VN</td></tr> <tr><td>23</td><td>EJ</td><td>VC30</td><td>VN</td></tr> <tr><td>24</td><td>EJ</td><td>VC31</td><td>VO</td></tr> <tr><td>25</td><td>EJ</td><td>VC32</td><td>VP</td></tr> <tr><td>26</td><td>EJ</td><td>VC33</td><td>VQ</td></tr> <tr><td>27</td><td>EJ</td><td>VC34</td><td>VQ</td></tr> <tr><td>30</td><td>EJ</td><td>VC35</td><td>VR</td></tr> <tr><td>31</td><td>EJ</td><td>VC36</td><td>VR</td></tr> <tr><td>32</td><td>EJ</td><td>VC37</td><td>VS</td></tr> <tr><td>33</td><td>EJ</td><td>VC56</td><td>VS</td></tr> <tr><td>34</td><td>MJ</td><td>O</td><td>VT</td></tr> <tr><td></td><td>CA</td><td>VE35</td><td></td></tr> </table>	0	IA	VE		1	TU	VA6	VE1	2	QT	VC13	A	3	EJ	VC13	VF	4	EJ	VC14	VF2	5	EJ	VC15	VH	6	EJ	VC17	VI	7	QT	VC7	A	10	EJ	VC42	VJ	11	EJ	VC43	VK	12	TP	Q	A	13	EJ	VC21	VJ	14	EJ	VC22	VL	15	EJ	VC40	VL	16	EJ	VC23	VM	17	EJ	VC24	VM	20	EJ	VC25	VM	21	EJ	VC26	VM	22	EJ	VC27	VN	23	EJ	VC30	VN	24	EJ	VC31	VO	25	EJ	VC32	VP	26	EJ	VC33	VQ	27	EJ	VC34	VQ	30	EJ	VC35	VR	31	EJ	VC36	VR	32	EJ	VC37	VS	33	EJ	VC56	VS	34	MJ	O	VT		CA	VE35		<table border="0"> <tr><td>Symbol</td><td>→</td><td>Q</td></tr> <tr><td>Ss. var. 77,76,75,</td><td>→</td><td>VF</td></tr> <tr><td>Single operand</td><td>→</td><td>VF2</td></tr> <tr><td>LIB</td><td>→</td><td>VH</td></tr> <tr><td>POW'S</td><td>→</td><td>VI</td></tr> <tr><td>(</td><td>→</td><td>VJ</td></tr> <tr><td>)</td><td>→</td><td>VK</td></tr> <tr><td>CW</td><td>→</td><td>A</td></tr> <tr><td> (open)</td><td>→</td><td>VJ</td></tr> <tr><td>(closed)</td><td>→</td><td>VL</td></tr> <tr><td>+</td><td>→</td><td>VM</td></tr> <tr><td>-</td><td>→</td><td>VM</td></tr> <tr><td>Unary -</td><td>→</td><td>VN</td></tr> <tr><td>,</td><td>→</td><td>VO</td></tr> <tr><td>=</td><td>→</td><td>VP</td></tr> <tr><td>*</td><td>→</td><td>VQ</td></tr> <tr><td>/</td><td>→</td><td>VR</td></tr> <tr><td>POW</td><td>→</td><td>VS</td></tr> <tr><td>Assume Δ.</td><td></td><td></td></tr> </table>	Symbol	→	Q	Ss. var. 77,76,75,	→	VF	Single operand	→	VF2	LIB	→	VH	POW'S	→	VI	(→	VJ)	→	VK	CW	→	A	(open)	→	VJ	(closed)	→	VL	+	→	VM	-	→	VM	Unary -	→	VN	,	→	VO	=	→	VP	*	→	VQ	/	→	VR	POW	→	VS	Assume Δ.		
0	IA	VE																																																																																																																																																																																	
1	TU	VA6	VE1																																																																																																																																																																																
2	QT	VC13	A																																																																																																																																																																																
3	EJ	VC13	VF																																																																																																																																																																																
4	EJ	VC14	VF2																																																																																																																																																																																
5	EJ	VC15	VH																																																																																																																																																																																
6	EJ	VC17	VI																																																																																																																																																																																
7	QT	VC7	A																																																																																																																																																																																
10	EJ	VC42	VJ																																																																																																																																																																																
11	EJ	VC43	VK																																																																																																																																																																																
12	TP	Q	A																																																																																																																																																																																
13	EJ	VC21	VJ																																																																																																																																																																																
14	EJ	VC22	VL																																																																																																																																																																																
15	EJ	VC40	VL																																																																																																																																																																																
16	EJ	VC23	VM																																																																																																																																																																																
17	EJ	VC24	VM																																																																																																																																																																																
20	EJ	VC25	VM																																																																																																																																																																																
21	EJ	VC26	VM																																																																																																																																																																																
22	EJ	VC27	VN																																																																																																																																																																																
23	EJ	VC30	VN																																																																																																																																																																																
24	EJ	VC31	VO																																																																																																																																																																																
25	EJ	VC32	VP																																																																																																																																																																																
26	EJ	VC33	VQ																																																																																																																																																																																
27	EJ	VC34	VQ																																																																																																																																																																																
30	EJ	VC35	VR																																																																																																																																																																																
31	EJ	VC36	VR																																																																																																																																																																																
32	EJ	VC37	VS																																																																																																																																																																																
33	EJ	VC56	VS																																																																																																																																																																																
34	MJ	O	VT																																																																																																																																																																																
	CA	VE35																																																																																																																																																																																	
Symbol	→	Q																																																																																																																																																																																	
Ss. var. 77,76,75,	→	VF																																																																																																																																																																																	
Single operand	→	VF2																																																																																																																																																																																	
LIB	→	VH																																																																																																																																																																																	
POW'S	→	VI																																																																																																																																																																																	
(→	VJ																																																																																																																																																																																	
)	→	VK																																																																																																																																																																																	
CW	→	A																																																																																																																																																																																	
(open)	→	VJ																																																																																																																																																																																	
(closed)	→	VL																																																																																																																																																																																	
+	→	VM																																																																																																																																																																																	
-	→	VM																																																																																																																																																																																	
Unary -	→	VN																																																																																																																																																																																	
,	→	VO																																																																																																																																																																																	
=	→	VP																																																																																																																																																																																	
*	→	VQ																																																																																																																																																																																	
/	→	VR																																																																																																																																																																																	
POW	→	VS																																																																																																																																																																																	
Assume Δ.																																																																																																																																																																																			

Subscripted Variable

	IA	VF			
0	TP	VC	VY2	}	0 (→ lists
1	RJ	VY	VY1		
2	RJ	VZ	VZ1		Subscripted variable to Processed List
3	MJ	0	XA		→ α
	CA	VF4			

Library Routine

	IA	VH			
0	TP	VA	Q	}	LIB? → VH12
1	QT	VA3	A		
2	ZJ	VH12	VH3		No ↓
3	QS	VC55	VA3		Set LIB
4	TP	VC46	VY2	}	LIB (→ lists
5	RJ	VY	VY1		
6	RJ	VZ	VZ1		LIB → Pro. List
7	TP	VC	VY2	}	0 (→ lists
10	RJ	VY	VY1		
11	MJ	0	XA		→ α
12	TP	VC	VY2		0 (→ lists
13	MJ	0	VH5		→ 40
	CA	VH14			

Special POWS

	IA	VI		
0	TP	VA	Q	} POW this level? VI51
1	QT	VA4	A	
2	ZJ	VI51	VI3	No ↓
3	QS	VC55	VA4	Set POW
4	QT	VA3	A	LIB? → VI34
5	ZJ	VI34	VI6	No ↓
6	TP	VA11	A	} Add 1 to count of last open
7	ST	VC1	A	
10	TU	A	VI11	} Add 1 to count of last open
11	TU	(30000)	VI12	
12	RA	(30000)	VC4	}) (count of one) → Pro. List
13	TV	VA7	VI14	
14	TP	VC5	(30000)	Increase add. of Pro. List
15	RA	VA7	VC1	} POW (→ (list
16	TV	VA11	VI20	
17	TP	VC47	A	(list +1
20	AT	VA	(30000)	} Add. of POW (→ (list
21	RA	VA11	VC1	
22	TV	VA11	VI23	(list + 1
23	TU	VI12	(30000)	Sq. sqrt. etc. → Pro. List
24	RA	VA11	VC1	Exit
25	RJ	VZ	VZ1	} Close to POW (no clear)
26	RJ	VI26	VI27	
27	TP	VC	VW2	} → (a)
30	SP	VC47	O	
31	AT	VA	VW3	LIB
32	RJ	VW	VW1	} Close to LIB (clear)
33	MJ	O	XA	
34	TP	VC20	VW2	} Change LIB to POW
35	TP	VC46	A	
36	AT	VA	VW3	(List +2
37	RJ	VW	VW1	} Clear LIB
40	TV	VA11	VI42	
41	TP	VC47	A	} Set print term
42	AT	VA	(30000)	
43	RA	VA11	VC2	→ (26)
44	TN	VA	Q	} Close to POW
45	QT	VA3	VA3	
46	TP	VA	Q	} → (27)
47	QS	VC55	VA2	
50	MJ	O	VI25	} Close to POW
51	TP	VC	VW2	
52	TP	VC47	A	} → (27)
53	AT	VA	VW3	
54	RJ	VW	VW1	} → (27)
55	MJ	O	VI46	
	CA	VI56		

(26)

(27)

Open Parenthesis (and Open Absolute |

	IA	VJ		
0	LQ	VA	1	Raise level
1	TP	VC44	VY2	} Add level and term ('s
2	RJ	VY	VY1	
3	TP	VC45	VY2	
4	RJ	VY	VY1	
5	MJ	0	XA	→ (α)
	CA	VJ6		

Closed)

	IA	VK		
0	RJ	XB	XB1	Print term checker
1	RJ	VB	VB1	Clear ind.
2	TP	VC20	VW2	} Close to level (clear) plus lower level
3	TP	VC44	A	
4	AT	VA	VW3	
5	RJ	VW	VW1	
6	LQ	VA	43	Lower level
7	MJ	0	XA	→ (α)
	CA	VK10		

Closed Absolute Value |

	IA	VL		
0	RJ	XB	XB1	→ Amb. term check
1	RJ	VB	VB1	Clear ind.
2	RJ	NT	NT1	Sym → Pro. List
3	MJ	O	VK2	Close to level (clear)
	CA	VL4		

	IA	NT		
0	MJ	O	30000	Exit
1	TP	VC	VW2	} Close to level (no clear)
2	TP	VC44	A	
3	AT	VA	VW3	
4	RJ	VW	VW1	
5	RJ	VZ	VZ1	Sym → Processed List
6	MJ	O	NT	Exit
	CA	NT7		

+ or -

	IA	VM		
0	RJ	XB	XB1	→ Ambiguous term checker
1	RJ	VB	VB1	Clear
2	RJ	NT	NT1	Close to level (no clear) sym → Pro.
3	TP	VC45	VY2	} Term (→ list
4	RJ	VY	VY1	
5	MJ	O	XA	→ (a)
	CA	VM6		

Unary Minus

	IA	VN		
0	TP	VC	VY2	}
1	RJ	VY	VY1	
2	RJ	VZ	VZ1	
3	RJ	VY	VY1	
4	MJ	O	XA	
	CA	VN5		

0 (→ lists
 - → Pro. List
 0 (→ lists
 → (a)

Comma

	IA	VO		
0	RJ	XB	XB1	}
1	TP	VC20	VW2	
2	TP	VC44	A	}
3	AT	VA	VW3	
4	RJ	VW	VW1	}
5	TP	VC44	VY2	
6	RJ	VY	VY1	}
7	TP	VC45	VY2	
10	RJ	VY	VY1	
11	RJ	VB	VB1	
12	MJ	O	XA	
	CA	VO13		

Amb. term checker
 Close to level (clear)
 Add level & term ('s
 Clear
 → (a)

Equals (=)

	IA	VP		
0	TP	VC20	VW2	} Close to level (clear)
1	TP	VC44	A	
2	AT	VA	VW3	
3	RJ	VW	VW1	} Add level & term ('s
4	TP	VC44	VY2	
5	RJ	VY	VY1	
6	TP	VC45	VY2	
7	RJ	VY	VY1	
10	MJ	O	XA	→ (a)
	CA	VP11		

Floating and Fixed *

	IA	VQ			
	0	TP	VA	Q	} LIB ↓ no → (30)
	1	QT	VA3	A	
	2	ZJ	VQ3	VQ4	} Set print term
(30)	3	QS	VC55	VA2	
	4	TN	VA	Q	
	5	RJ	VB	VB3	} Clear LIB, POW, /
(19)	6	TP	VC	VW2	
	7	TP	VC45	A	} Close to term (no clear)
	10	AT	VA	VW3	
	11	RJ	VW	VW1	
(36)	12	RJ	VZ	VZ1	* → Pro. List
	13	TP	VC	VY2	0 (→ lists
	14	RJ	VY	VY1	
	15	MJ	O	XA	→ (a)
		CA	VQ16		

Floating and Fixed /

	IA	VR		
0	TP	VA	Q	} /→ VR6 no↓
1	QT	VA5	A	
2	ZJ	VR6	VR3	
3	QS	VC55	VA5	Set /
4	QT	VA3	A	} LIB↓ no → 33
5	ZJ	VR6	VR7	
6	QS	VC55	VA2	Set print term
7	TN	VA	Q	} Clear LIB & POW
10	QT	VA3	VA3	
11	QT	VA4	VA4	
12	MJ	0	VQ6	→ 19
	CA	VR13		

32
33

POW

	IA	VS		
0	RJ	VI26	VI	→ POW sect.
1	MJ	0	XA	→ α
	CA	VS2		

Space Period Δ .

	IA	VT		
0	TP	VA16	Q	} Print term ↓ no → VT5
1	QJ	VT2	VT5	
2	RJ	WA	WA2	} Print WARNING, ΔΔ AMBIGUOUS TERMS.
3	TP	XC	UP3	
4	RJ	UP2	UP	
5	TP	VC20	VW2	
6	TP	VC44	A	
7	AT	VA	VW3	} Close to level (clear)
10	RJ	VW	VW1	} Clear
11	RJ	VB	VB1	
12	RJ	VZ	VZ1	Δ . → Pro. List
13	TP	VC60	VA6	Set address of no. list
14	MJ	O	VU	→ numbering routine
	CA	VT15		

Print

	IA	XC		
0	40	XC1	5	
1	71	24545	03450	W A R N I N
2	32	21010	12447	G , Δ Δ A M
3	25	34326	75167	B I G U O U
4	65	01663	05447	S Δ T E R M
5	17	65432	27777	(S) . 77 77
	CA	XC6		

Numbering Routine

	IA	VU		
0	TP	VC6	VA14	Set (no. =1
1	RS	VA7	VC1	} Finished numbering → SR No ↓
2	TJ	VC61	SR	
3	TU	A	VU4	
4	TP	(30000)	Q	} Sym → Q
5	QT	VC7	A	
6	EJ	VC42	VU16	(→ VU16
7	EJ	VC43	VU23)→ VU23
10	TU	VA10	VU11	} No. → A
11	TP	(30000)	A	
12	TV	VA6	VU13	} No. sym → list
13	AT	Q	(30000)	
14	RA	VA6	VC1	Address +1
15	MJ	0	VU1	
16	QT	VC12	Q	} Count → Q
17	SP	Q	17	
20	AT	Q	VA15	} Take nos. off list
21	RS	VA10	VA15	
22	MJ	0	VU1	
23	QT	VC12	A	Count → A
24	ST	VC4	VA15	Set index
25	RA	VA10	VC1	} Add nos. to list
26	TV	VA10	VU27	
27	TP	VA14	(30000)	
30	RA	VA14	VC6	Increase highest no.
31	IJ	VA15	VU25	
32	MJ	0	VU1	
	CA	VU33		

Sort Routine

0	IA	SR			
1	SP	VA6	0	}	No. to be sorted → VA21
2	ST	VC60	VA21		
3	TP	VC7	Q	}	Set n of repeat
4	QS	VA21	SR4		
5	RP	30000	SR6	}	List negative
6	TN	WL24	WL24		
7	TP	VC60	VA6		Address of no. list = WL24
10	SP	VC61	0	}	Address of Sorted List → VA7
11	SA	VA21	0		
12	ST	VC1	VA7	}	1st sym → Sorted List
13	TV	A	SR13		
14	TP	WL24	(30000)		No. List +1
15	RA	VA6	VC1	}	# of nos. in Sorted List → VA22
16	TU	A	SR23		
17	TU	A	SR51	}	
20	ST	VC60	VA22		
21	TP	VC7	Q	}	Set n of repeat
22	QS	VA22	SR24		
23	TU	VA7	SR25		Set address of Sorted List
24	TP	(30000)	A		# → A
25	RP	20000	SR35	}	largest # yet → SR35 No ↓
26	TJ	(30000)	SR26		
27	TU	SR24	VA24		j n → VA24
30	LQ	Q	17	}	r - 1 → VA12
31	SP	VA24	0		
32	SS	Q	0	}	Set repeat to move back nos.
33	ST	VC3	VA12		
34	AT	VC52	SR43	}	Set to move back all nos.
35	MJ	0	SR40		
36	TP	VC7	Q	}	r - 1 = all nos.
37	QS	SR24	SR43		
40	SP	VC54	0		TP 0 0
41	SA	VA7	0		TP SL+ SL+
42	ST	VC4	SR44		TP SL+ (SL+) -1
43	RP	30000	SR45	}	Move nos. back
44	TP	(30000)	(30000)		
45	LQ	VA12	25		r - 1 → V address
46	SP	SR44	0	}	TP no. L+ (SL+) + r - 1
47	SA	Q	0		
50	TV	A	SR51	}	Sorted list address -1
51	TP	(30000)	(30000)		
52	RS	VA7	VC1		Done → SR55 no ↓
53	TJ	VC50	SR55		→ SR14
54	MJ	0	SR14		
55	TU	SR4	SR56		Set n of repeat

56	RP	0	VD	}	Exit
57	TN	PR	WL4		Change to positive
	CA	SR60			

Add Parenthesis to Lists

	IA	VY		
0	MJ	0	(30000)	Exit
1	MJ	0	VY3	Start
2	0	0	0	Type of (to add (no level)
3	TV	VA7	VY4	} (→ Processed List
4	TP	VC42	(30000)	
5	TV	VA11	VY7	} Code word→ (list (count zero)
6	TP	VY2	A	
7	AT	VA	(30000)	
10	RA	VA11	VC1	} Address→ (list
11	TV	VA11	VY12	
12	TU	VA7	(30000)	
13	RA	VA7	VC1	Pro. List +1
14	RA	VA11	VC1	(list + 1
15	MJ	0	VY	Exit
	CA	VY16		

Sym → Processed List

	IA	VZ		
0	MJ	0	(30000)	Exit
1	TU	VA6	VZ3	
2	TV	VA7	VZ3	Sym→ Pro. List
3	TP	(30000)	(30000)	
4	RA	VA7	VC1	Pro. List +1
5	MJ	0	VZ	Exit
	CA	VZ6		

Close Parentheses

	IA	VW		
0	MJ	0	(30000)	Exit
1	MJ	0	VW4	Start
2	0	0	0	- Take off list, + leave on
3	0	0	0	Code of (and level
4	TV	VA7	VW5	}) → Pro. List (count zero)
5	TP	VC43	(30000)}	
6	TU	VA11	VW10}	} Code of (→ A
7	RS	VW10	VC57}	
10	TP	(30000)	A	
11	EJ	VW3	VW15	= → VW15 no ↓
12	RJ	VX	VX1	(+1 and) + 1
13	RS	VA11	VC2	Take (off list
14	MJ	0	VW6	Return
15	RJ	VX	VX1	(+1 and) + 1
16	TP	VW2	Q	} Delete from list ↓ no → VW21
17	QJ	VW20	VW21}	
20	RS	VA11	VC2	Clear (from list
21	RA	VA7	VC1	Add. of Pro. List +1
22	MJ	0	VW	Exit
	CA	VW23		

(+1 and) +1

	IA	VX		
0	MJ	0	(30000)	Exit
1	TU	VW10	VX3	} Increase count on open
2	RA	VX3	VC3	
3	TU	(30000)	VX4	
4	RA	(30000)	VC4	
5	TU	VA7	VX6	} Increase count on closed
6	RA	(30000)	VC4}	
7	MJ	0	VX	Exit
	CA	VX10		

Clear Indicators

	IA	VB					
0	MJ	0	(30000)	Exit			
1	TN	VA	Q				
2	QT	VA2	VA2	P.T.	}	clear	
3	QT	VA3	VA3	LIB			
4	QT	VA4	VA4	POW			
5	QT	VA5	VA5	DIVIDE			
6	MJ	0	VB	Exit			
	CA	VB7					

Check for Ambiguity

	IA	XB				
0	MJ	0	(30000)	Exit		
1	TP	VA	Q	}	Ambiguity ↓ no → exit	
2	QT	VA2	A			
3	ZJ	XB4	XB			
4	TP	VC20	VA16	Set indicator		
5	MJ	0	XB	Exit		
	CA	XB6				

Constants

	IA	VC			
0	0	0	0	0	Zero
1	0	1	1	1	One
2	0	2	2	2	Two
3	0	1	0		One in u
4	0	0	1		One in v
5	0	2	1) count of 1
6	0	1	0		(numbering bit
7	0	07777	0		Sort
10	0	0	0		NP routine
11	0	0	07777		
12	0	0	77777		
13	0	0	70000		Sub. var.
14	0	0	60000		Single operand
15	0	0	50000		LIB
16	0	0	40000		Pseudo Op.
17	0	0	10000		POW'S
20	40	0	0		Close off bit indicator
21	0	0	10		(open)
22	0	0	12		(closed) floating
23	0	0	20		F1. +
24	0	0	21		Fx. +
25	0	0	30		F1. -
26	0	0	31		Fx. -
27	0	0	32		F1. Unary -
30	0	0	33		Fx. Unary -
31	0	0	40		,
32	0	0	50		=
33	0	0	60		F1. *
34	0	0	61		Fx. *
35	0	0	70		F1. /
36	0	0	71		Fx. /
37	0	0	100		POW
40	0	0	13		(closed) fixed
41	0	0	120		Δ .
42	0	1	0		(
43	0	2	0)
44	1	0	0		Level
45	2	0	0		Term
46	3	0	0		LIB
47	4	0	0		POW
50	0	PR1	PR1		Sort
51	0	PR1000	PR1000		Limit of Processed List
52	RP	30000	SR45		Sort routine
53	0	0	2		NP3 and NP32
54	TP	0	0		Sort
55	77	77777	77777		

56	0	0	101	POW (int.)
57	0	2	0	2 in u
60	0	WL24	WL24	Sym/wd and No. Lists
61	0	PR	PR	Processed and Sorted Lists
62	0	NS	NS	Number of Symbol List
63	0	PL	PL	Parenthesis List
	CA	VC64		

Variables (VA) - Explanation of Temporaries

VA 0	0	0	1	Level bit
1	()		Combination List size
2				Print Term this level
3				LIB this level
4				POW this level
5				Divide this level
6	()	()	Address in Sym/Wd List and Numbered List
7	()	()	Address in Processed List and Sorted List
10	()	()	Address of Symbol Number
11	()	()	Available address in (list
12	()		r - 1 in sort
13	()		j n in print
14				Highest number of ('s
15				Index
16				Temp 1- ambiguous term bit
17	()		Temp 2
20	()		Temp 3- add. to start pr. of Pro. List
21		()	()	n of repeat to set Pro. List
22		()	()	# of nos. in Sorted List
23	0	0	(0)	Unused
24	()		j n in sort.

EQUATION GENERATION NO. 2

EQUATION REDUNDANCY CHECK AND EQUATION GENERATION PHASE

The purpose of the Equation Redundancy check and Equation Generation Phase is two-fold:

- 1) The elimination of redundant calculations within the same equation;
- 2) The generation of a relatively coded routine for each equation.

The inputs to this phase are the Sorted List, the Dimension List, and the Pseudo Operation List. The symbols for a given equation are obtained in order from the Sorted List and each operator, together with its operand (s), is put in the form of a pseudo instruction to facilitate the check for redundant calculations. These pseudo instructions are entered in what is called the Expanded List, unless an identical pseudo instruction has been previously entered. In the case of an identical previous entry, the current pseudo instruction represents a redundant calculation and provision is made to utilize the result of the prior calculation. Through the special formats for the pseudo instructions, many redundant calculations will be eliminated. For example:

- 1) Identical Symbol Strings.
eg., $X = \sin (A+B+C-D/E) + (A+B+C-D/E) \text{ Pow } 2$
The quantity $(A+B+C-D/E)$ will be calculated only once.
- 2) Simple Transpositions.
eg., $X = A*B - \sin(B*A)$
The quantity $A*B$ will be recognized as equivalent to the quantity $B*A$ and would not be recomputed.
- 3) Transpositions within Expressions where some reordering is caused by the hierarchy of operators.
eg., $X = (A+B*C)/E - \tan((C*B+A)/E)$
The quantities $(A+B*C)/E$ and $(C*B+A)/E$ will be recognized as equivalent and only one computation will be made.

A unique partial result symbol for each calculation is entered in the Expanded List following each pseudo instruction. This partial result symbol identifies the result of a given calculation as an operand for a succeeding calculation. When a partial result from a calculation is used as an operand for

the next calculation, register storage (A or Q) may be utilized; hence, each pseudo instruction is checked to determine if the last assigned partial result appears as one of its operands. In this way, effective utilization of register storage is realized; thereby minimizing the need for temporary storage.

The Expanded List, together with lists of supplemental information, serves as input for the generation of the relatively coded equation routine. Each pseudo instruction is obtained in order from the Expanded List and decoded. The series of relatively coded machine instructions necessary to perform the required computation and store the partial result is then generated. After all pseudo instructions have been processed, the fixed constants and relative constants are transferred to the generated routine package. At this time also, the Op File describing this generated routine is prepared. The equation routine and Op File are then transcribed on magnetic tape for use as input to succeeding phases of the compiler.

As an example, consider the equation:

$$X = A+B*C - \sin(C*B)$$

In the Sorted List this equation would appear as:

X
B
C
*
A
+
C
B
*
sin
-
Δ.

Following the elimination of redundant calculations, the equation appears in pseudo instruction form in the Expanded List as:

*	B	C
		PR 1
+	PR 1	A
		PR 2
sin	0	PR 1
		PR 3
-	PR 2	PR 3
		PR 4
Δ.	PR 4	X

Note 1: (PR__) represents unique partial result symbols.

Note 2: The computation of the quantity (C*B) is recognized as a redundant calculation and the result of the prior calculation (PR 1) is used as the argument for the "sin" operator.

The Expanded List is processed to form the following generated equation routine:

EXIT	MJ	0	[]	
ENTRY	FM	B	C	$B * C \rightarrow Q$
	TP	Q	TEMP 1	$B * C \rightarrow TEMP 1$
	FA	Q	A	$B * C + A \rightarrow Q$
	TP	Q	TEMP 2	$B * C + A \rightarrow TEMP 2$
	TP	TEMP 1	SIN	$B * C \rightarrow SIN + 3$
	10	0	3	
	RJ	SIN	SIN	$SIN(B * C) \rightarrow Q$
	10	2	0	
	TN	Q	Q	$-SIN(B * C) \rightarrow Q$
	FA	Q	TEMP 2	$[-SIN(B * C)] + [B * C + A] \rightarrow Q$
	TP	Q	X	$A + B * C - SIN(B * C) \rightarrow X$

Consider another equation which appears in the Sorted List as:

8	X
6	B
5	C
4	D
4	POW
3	*
2	A
2	+
1	Δ.

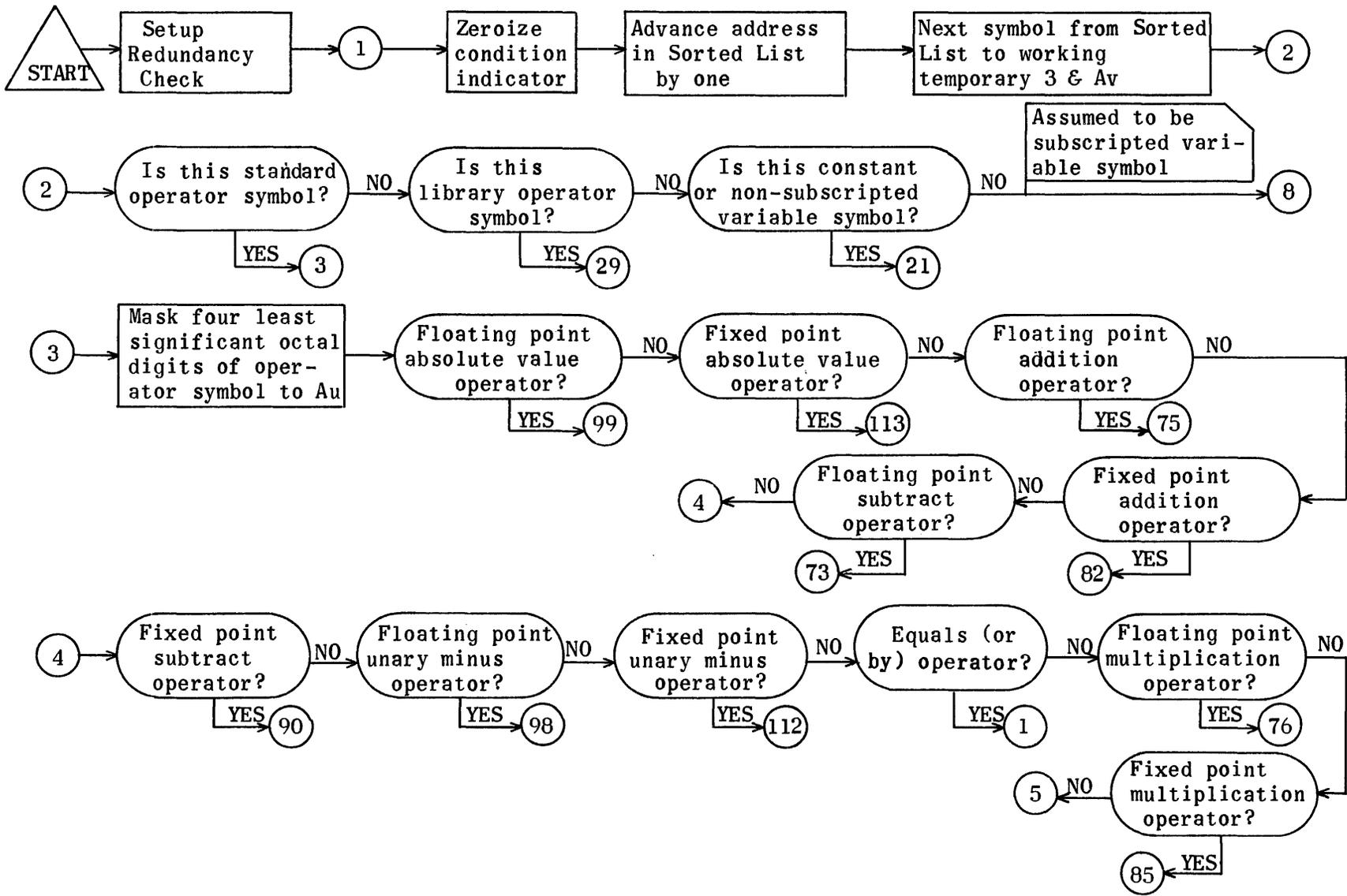
Following the elimination of redundancies (none in the example), the equation appears in the Expanded List as:

POW	C	D
*	PR 1	PR 1 B
+	PR 2	PR 2 A
Δ.	PR 3	PR 3 X

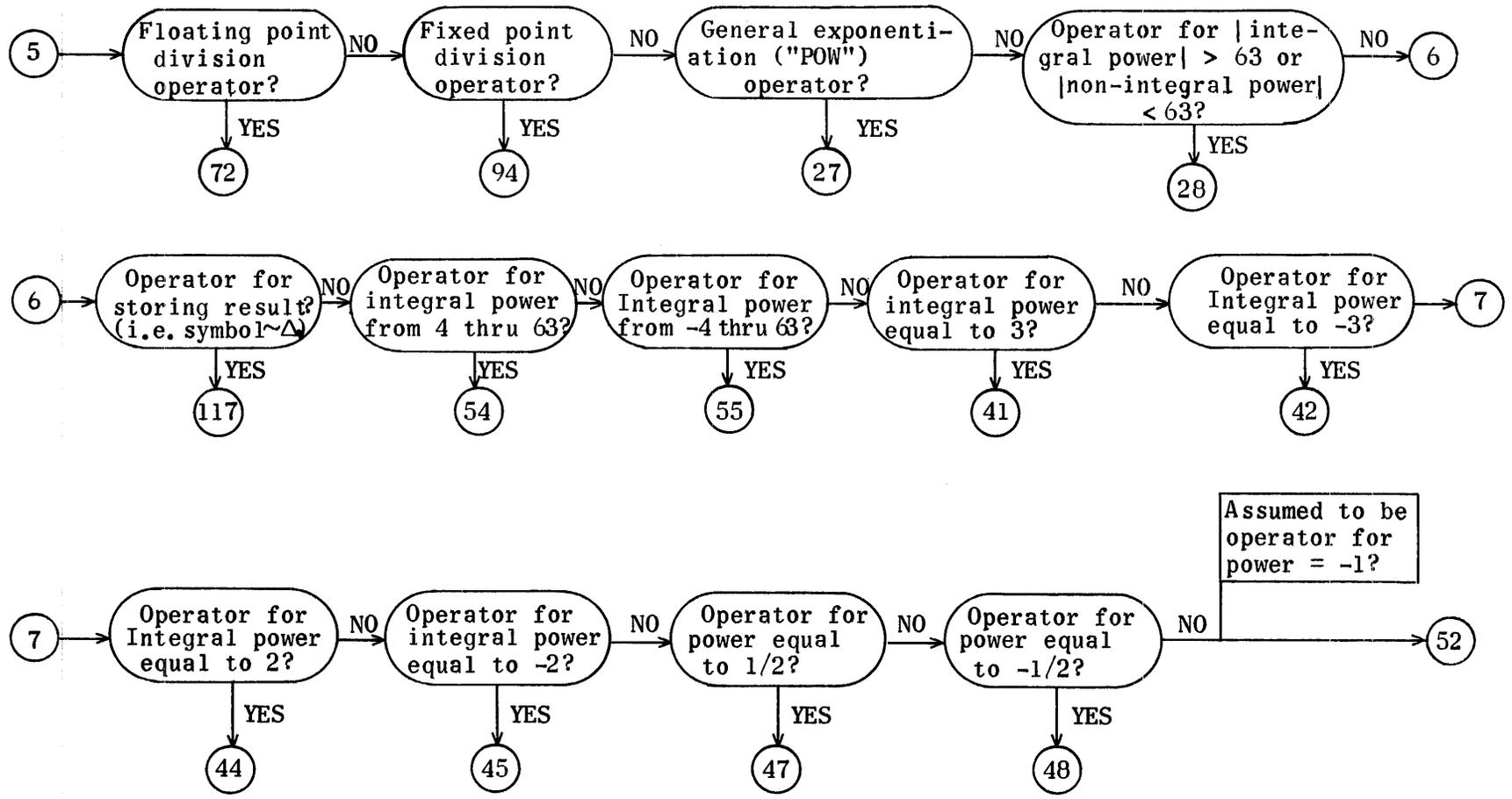
Finally, the generated equation routine would be:

EXIT	MJ	0	[]
ENTRY	TP	C	POW
	10	0	3
	TP	D	POW
	10	0	4
	RJ	POW	POW
	10	2	0
	FM	Q	B
	FA	Q	A
	TP	Q	X
	MJ	0	EXIT

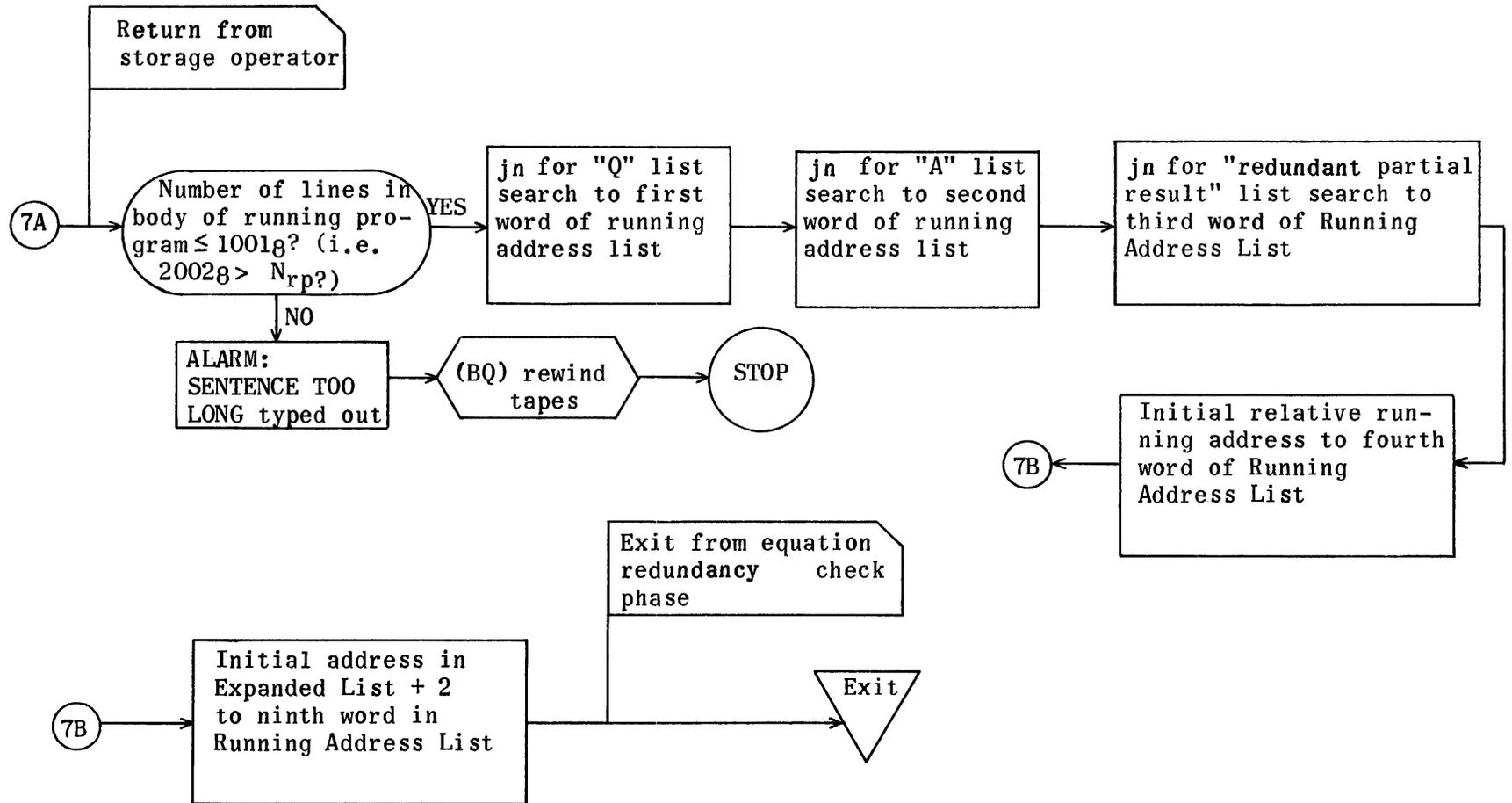
Equation Redundancy Check (Symbol Search)



1234

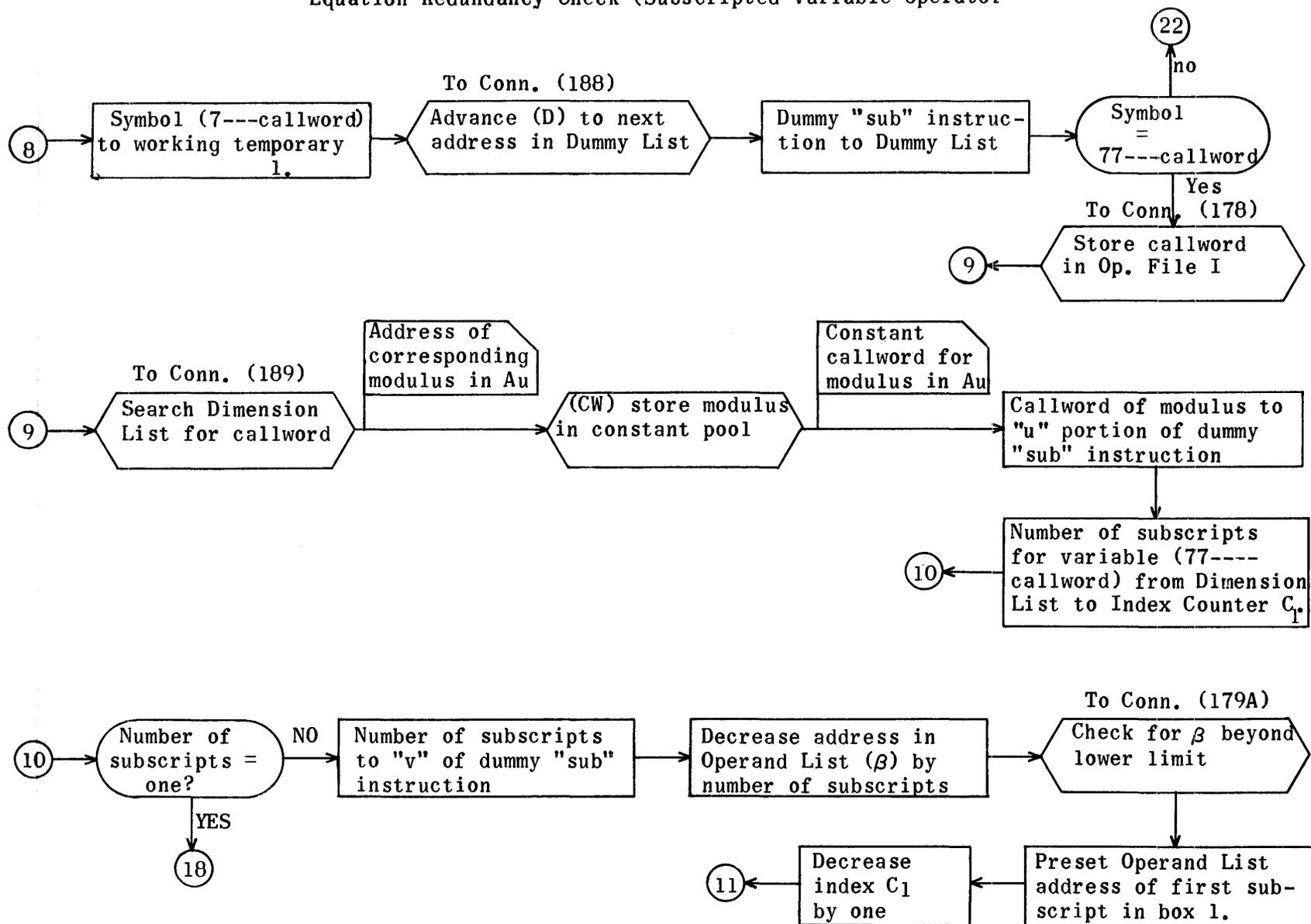


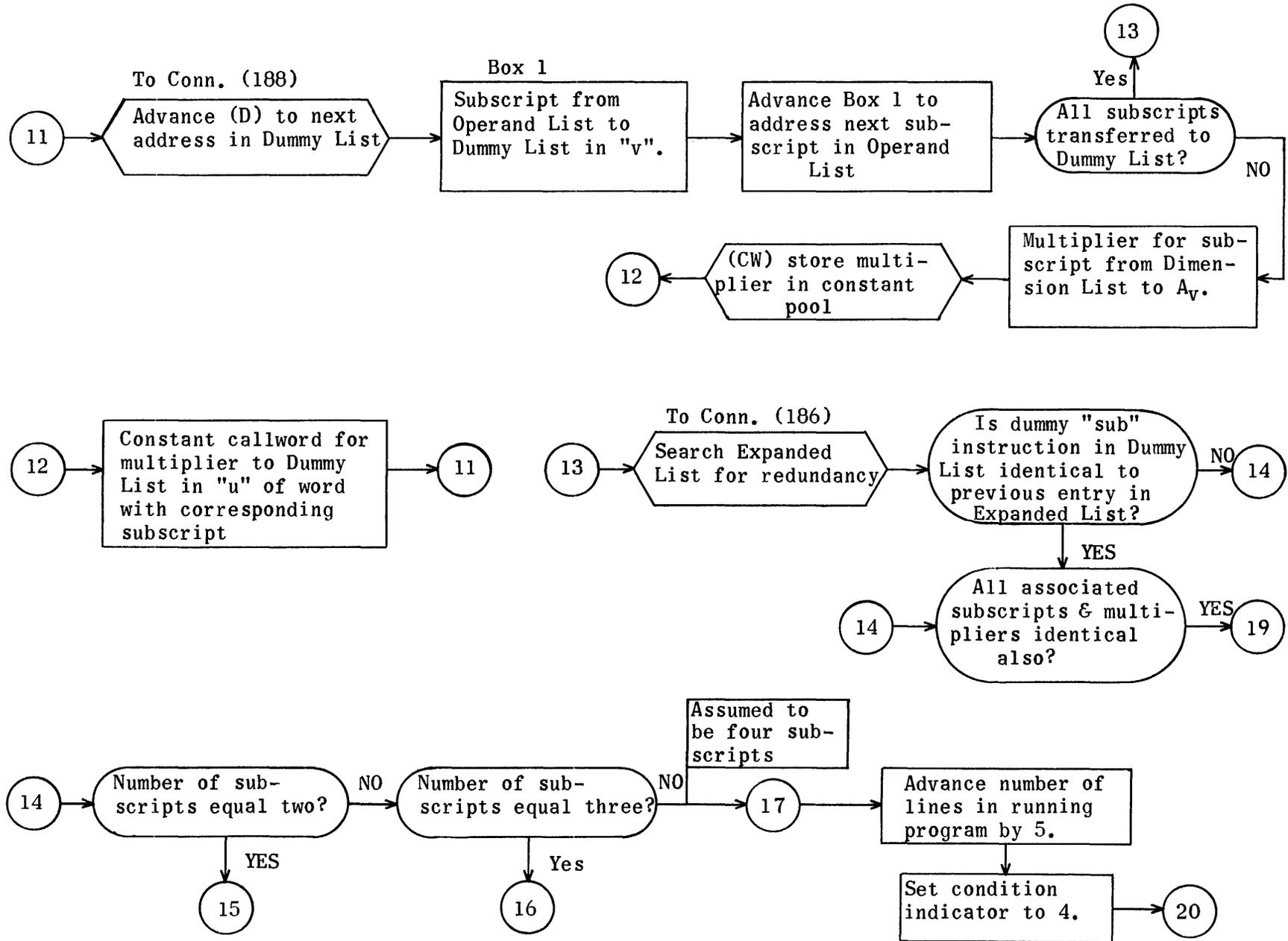
END EQUATION REDUNDANCY CHECK

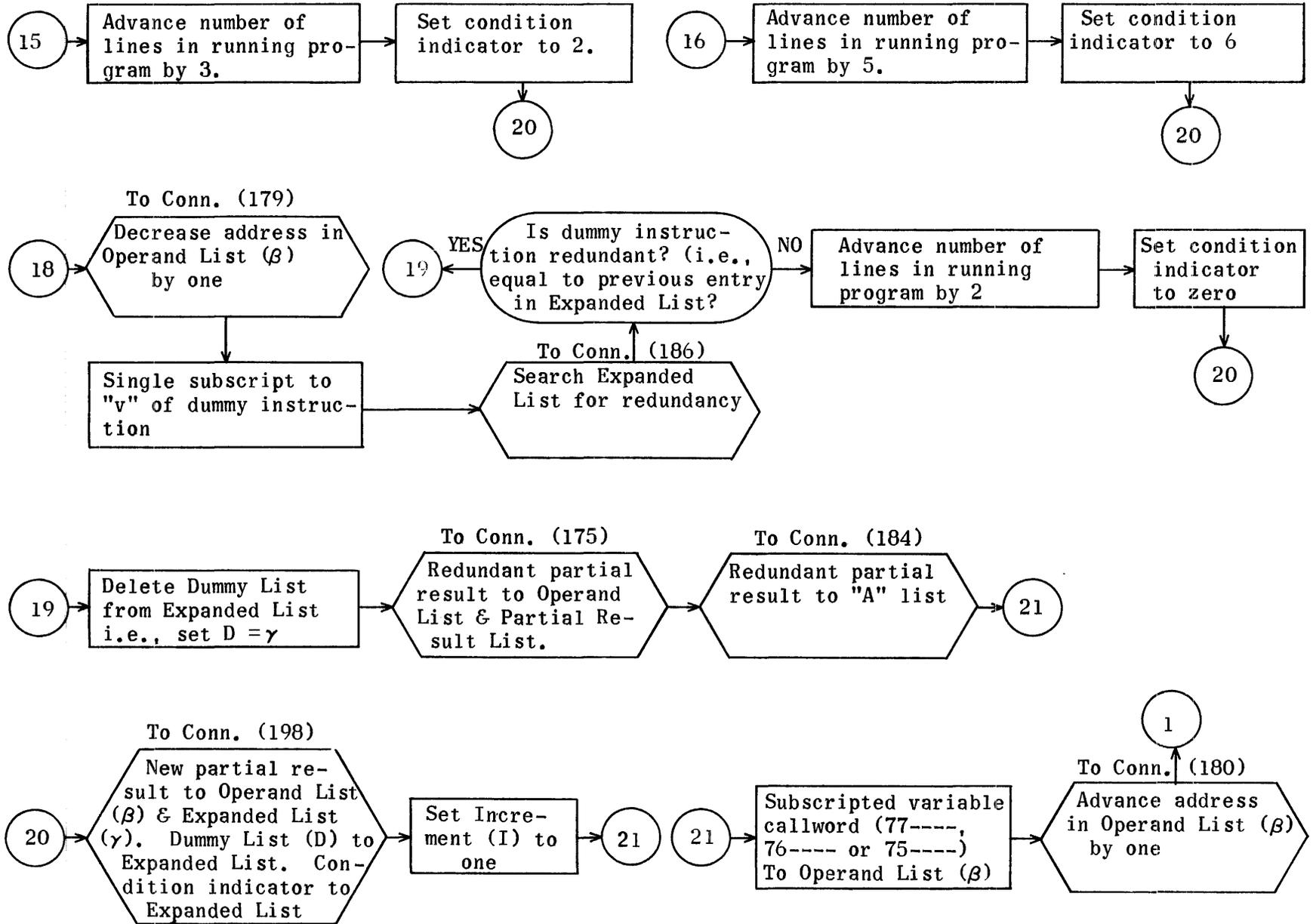


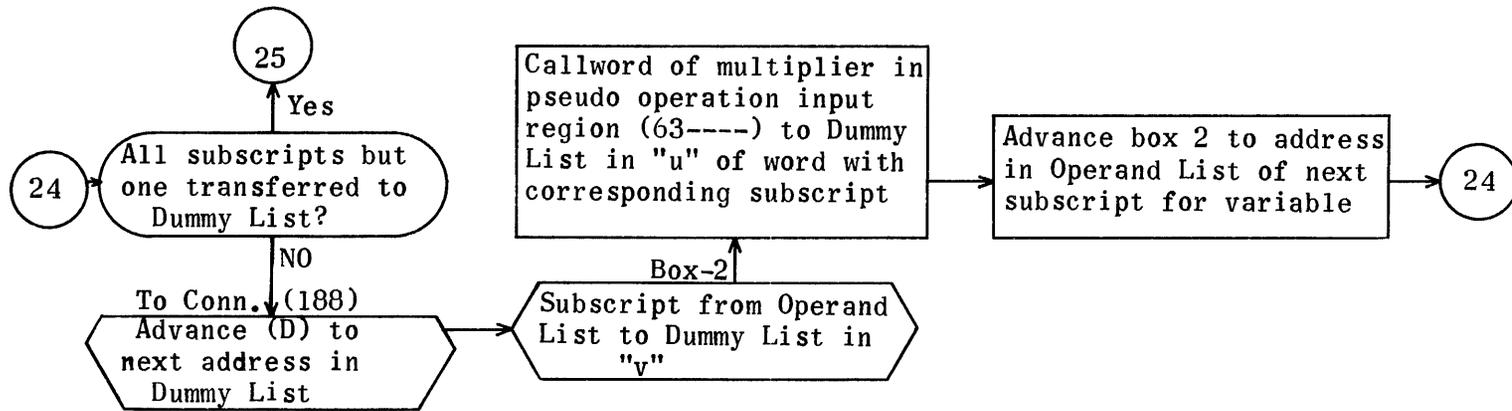
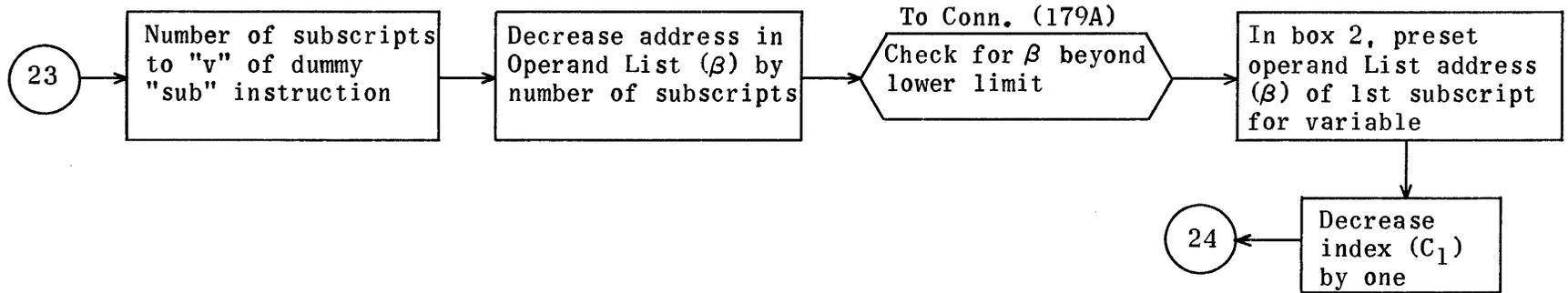
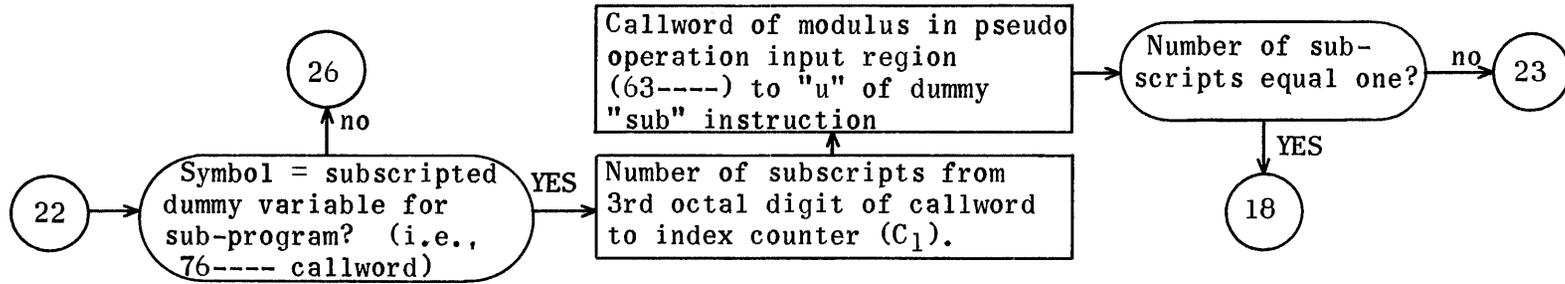
1236

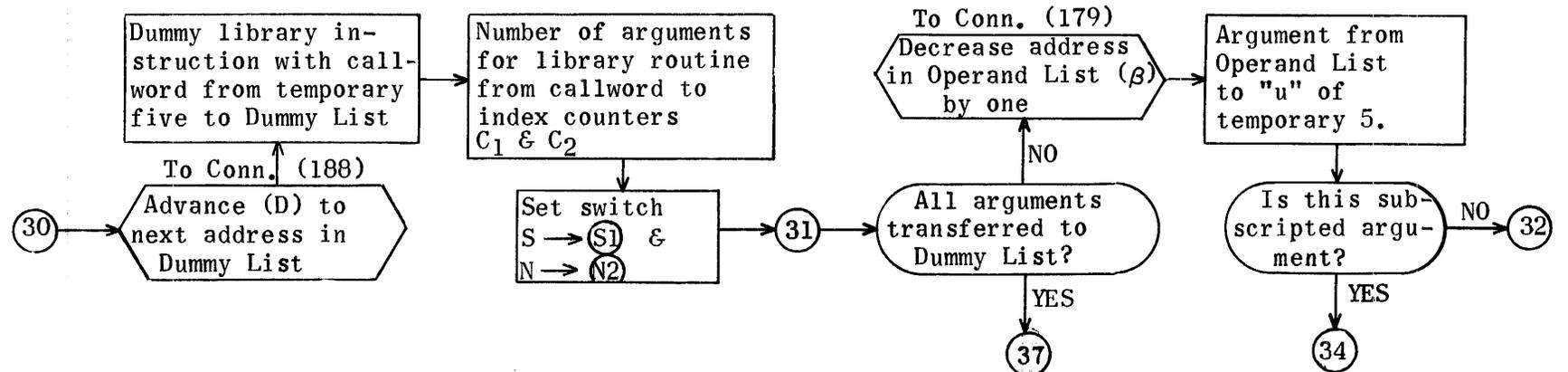
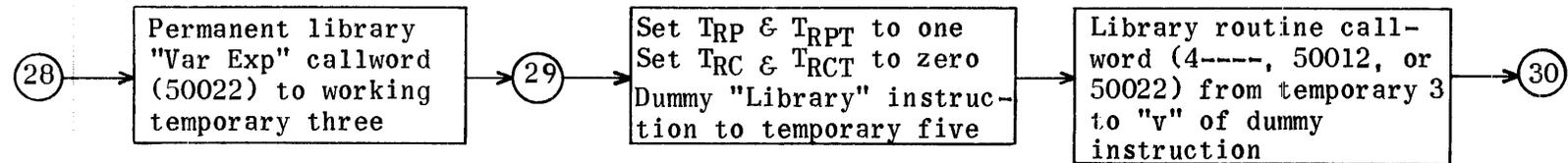
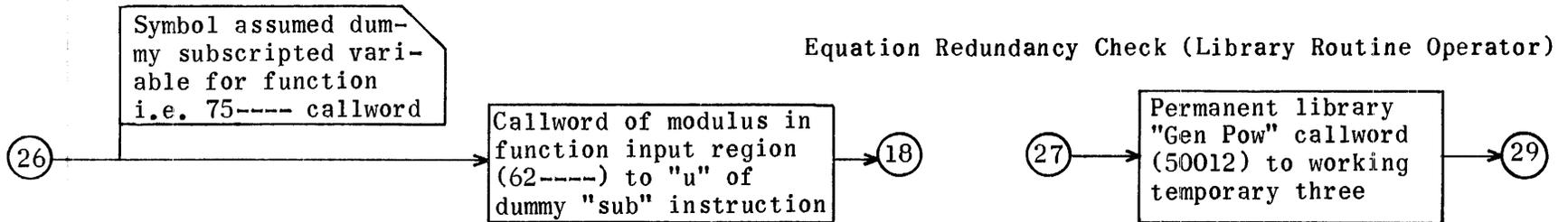
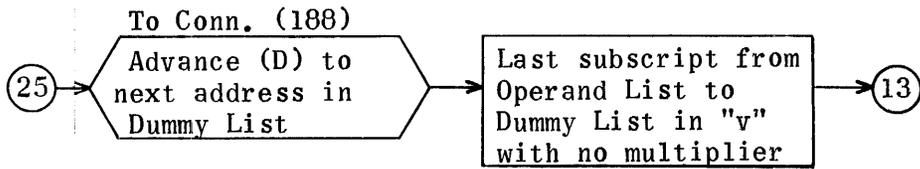
Equation Redundancy Check (Subscripted Variable Operator

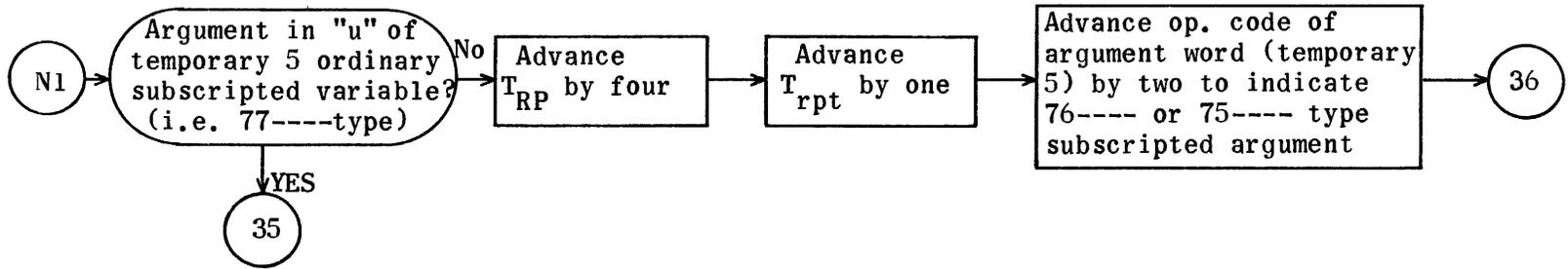
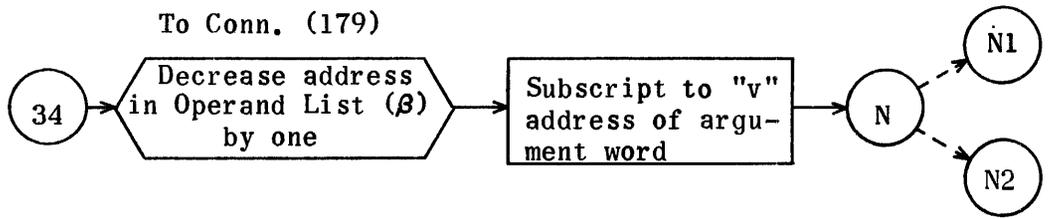
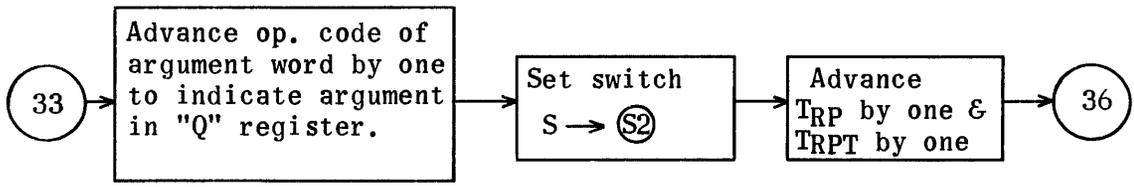
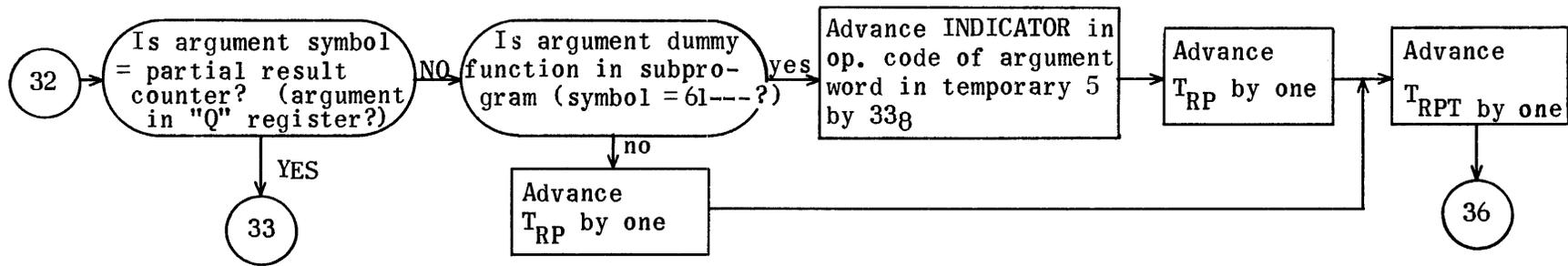






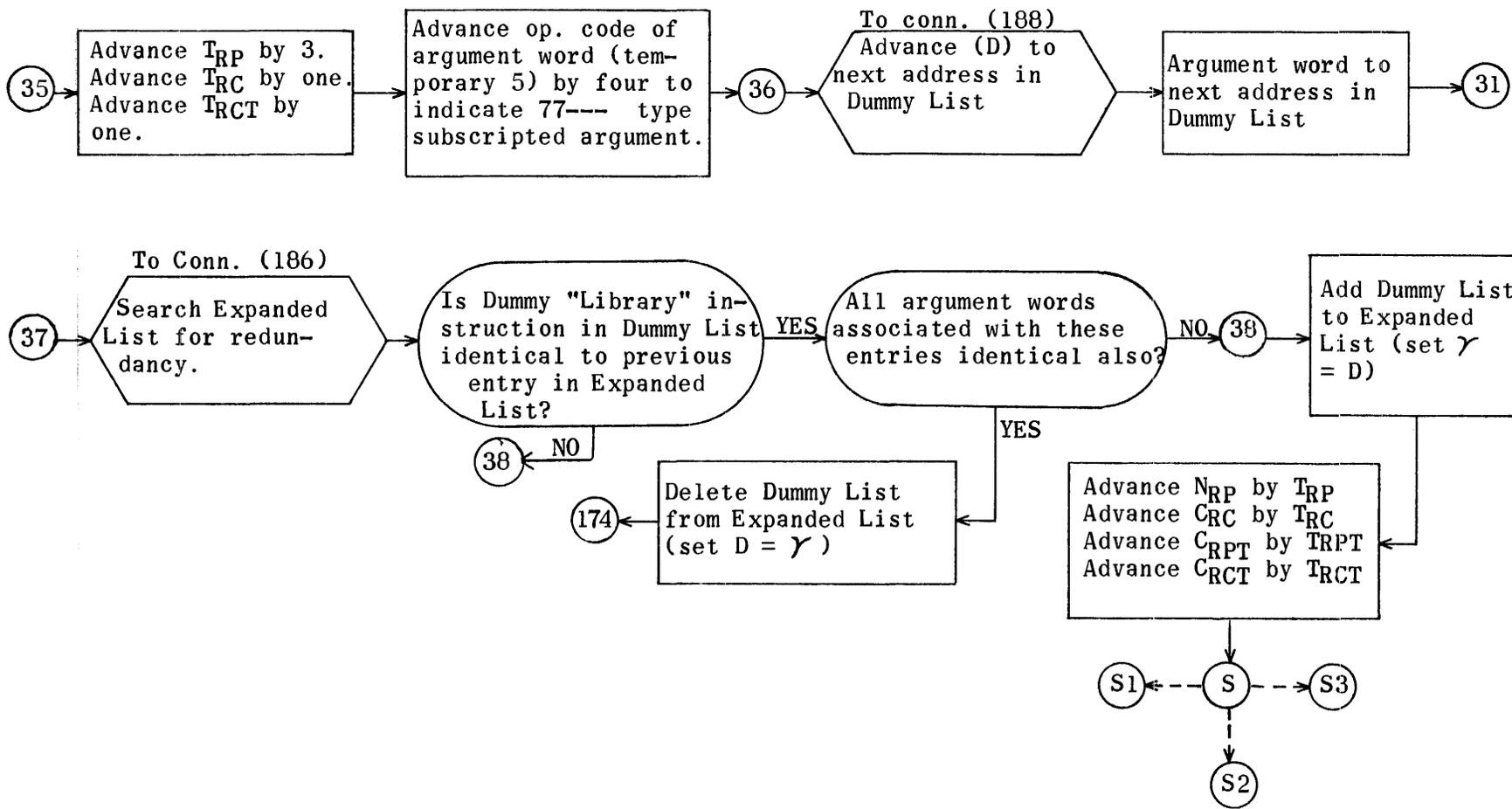




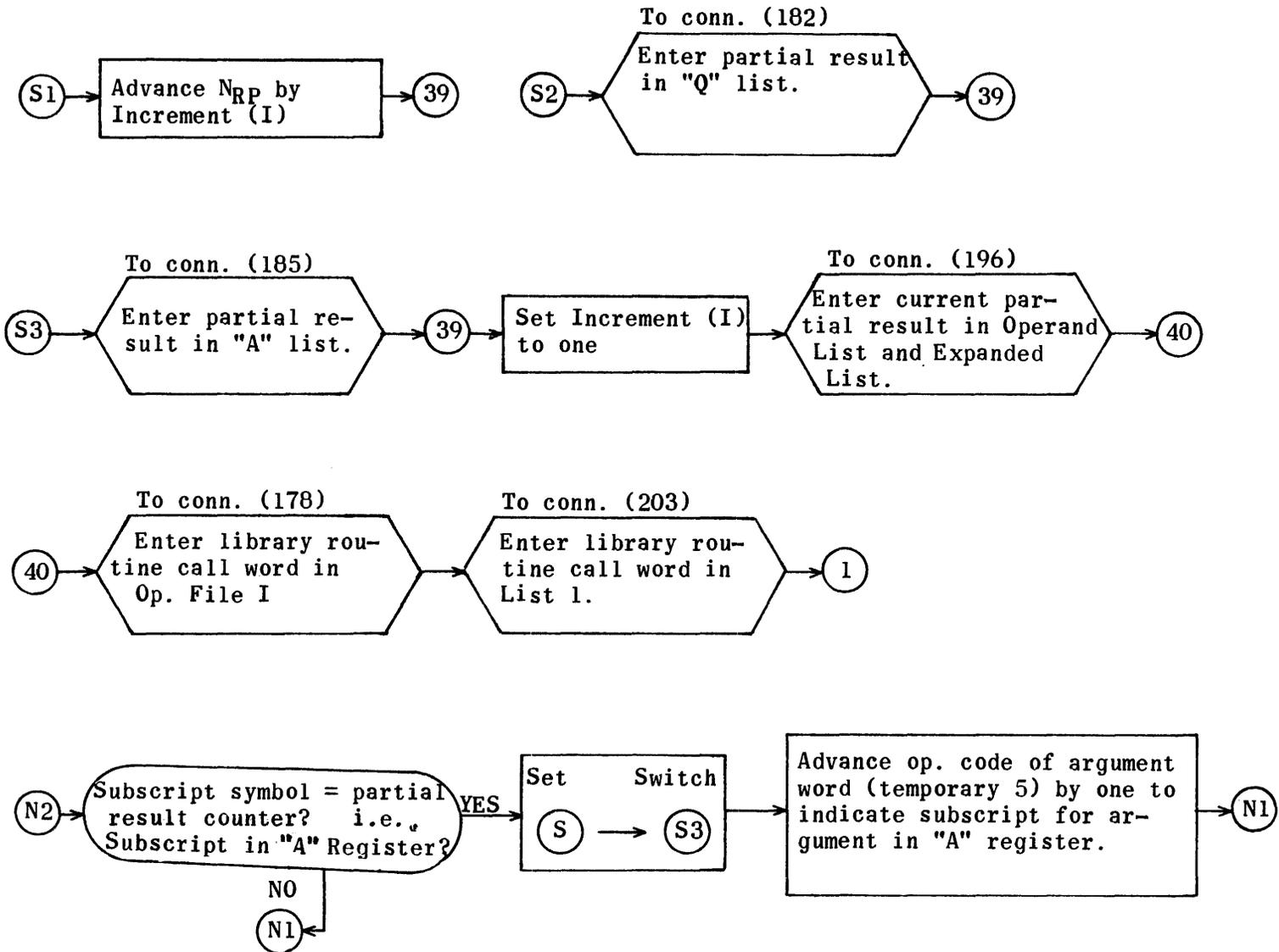


1242

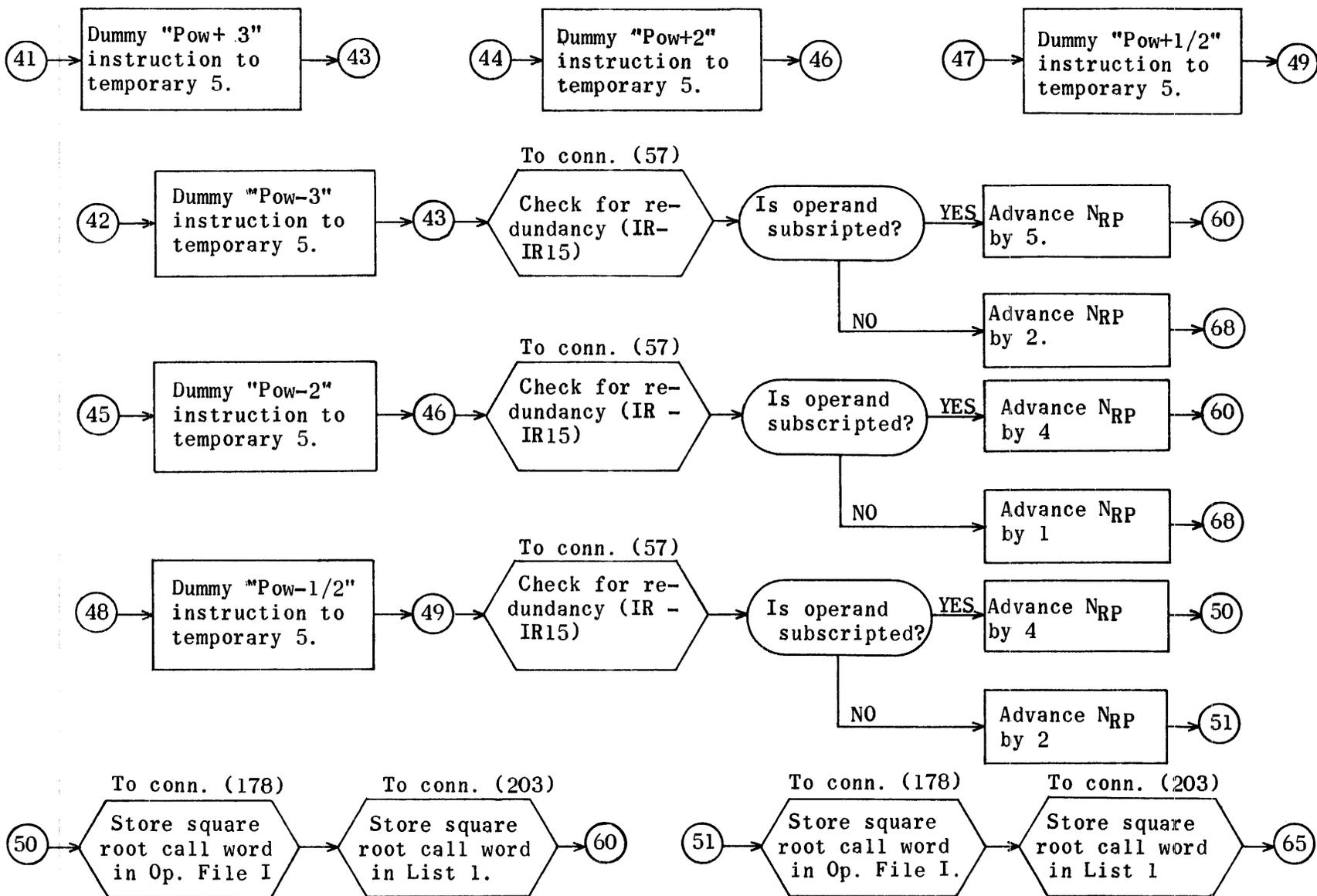
Equation Redundancy Check (Library Routine Operator)



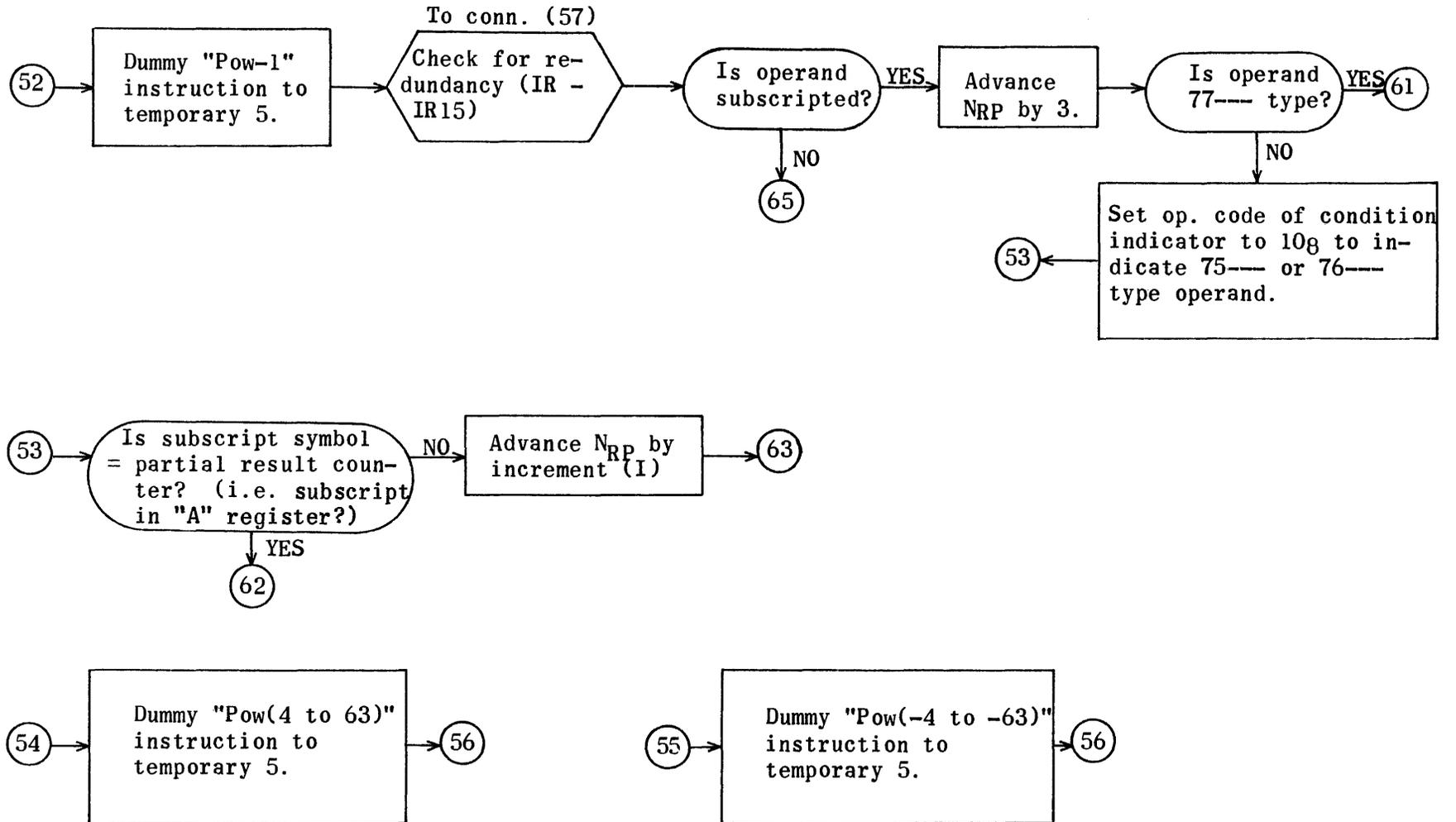
1243

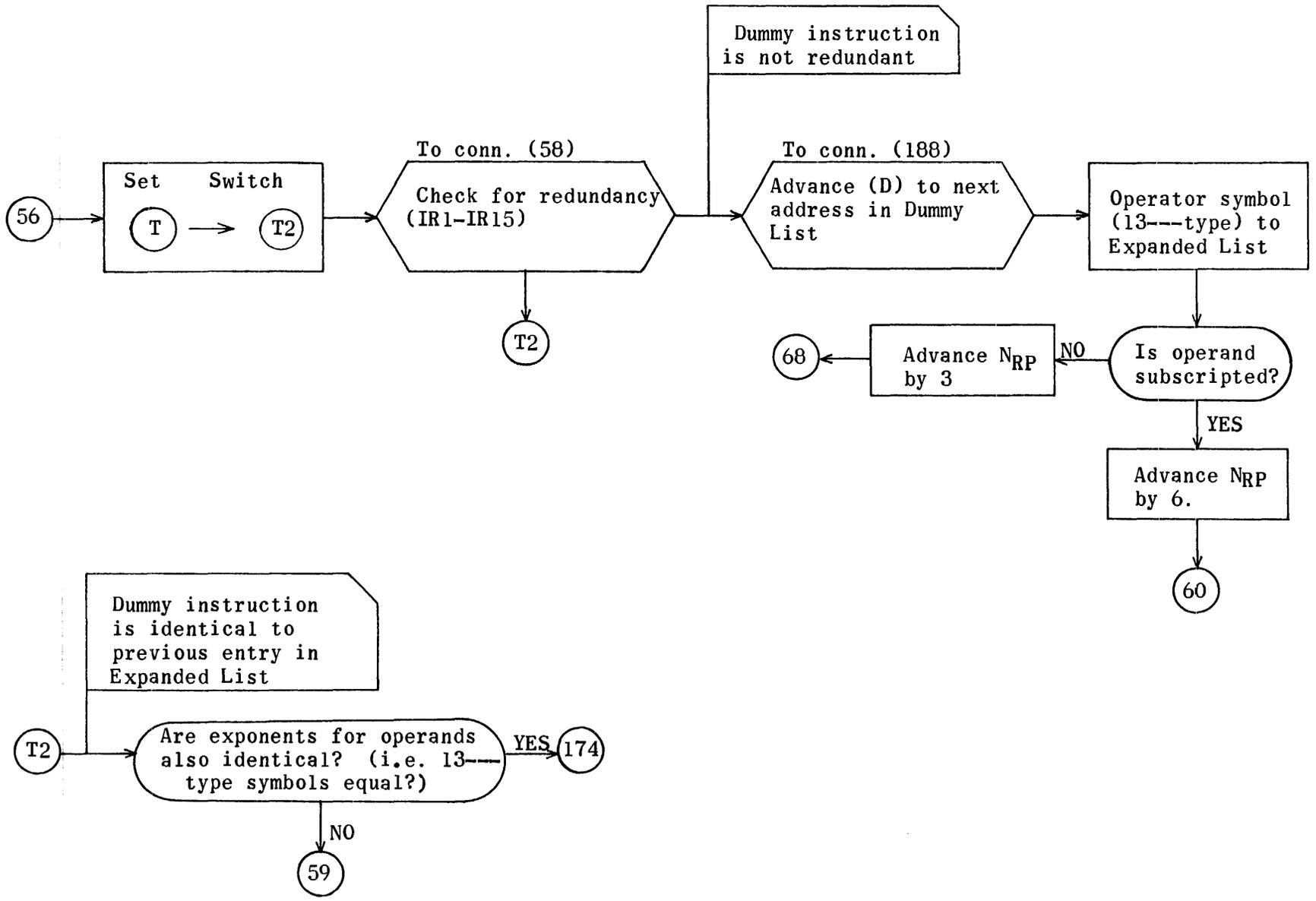


Equation Redundancy Check (Power Operators)

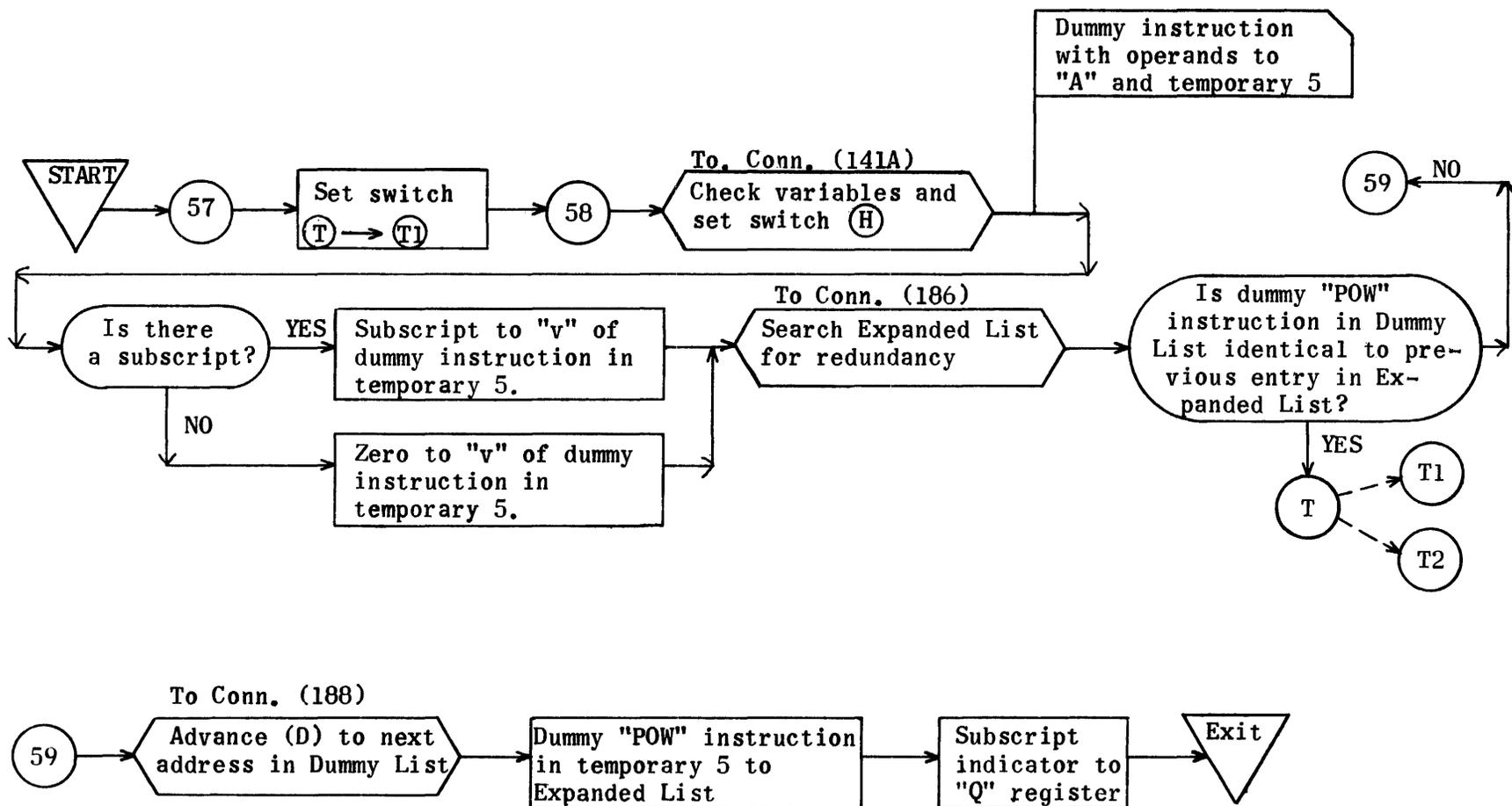


Equation Redundancy Check (Integral Power Operator)

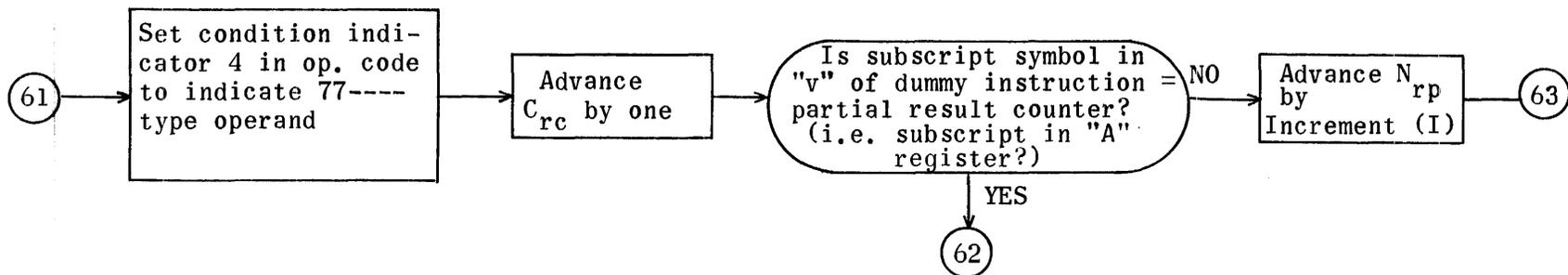
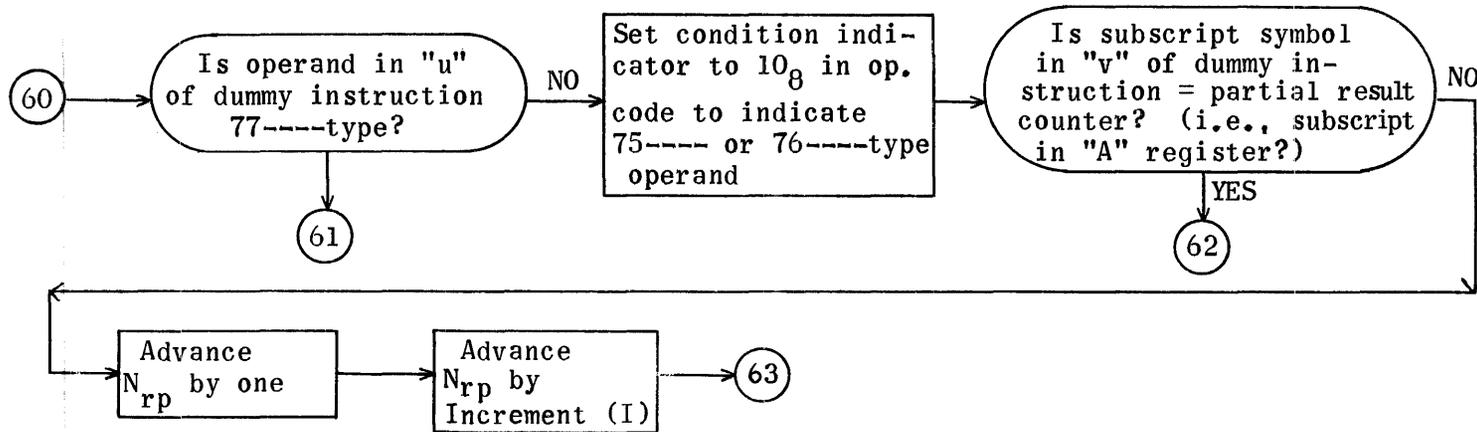


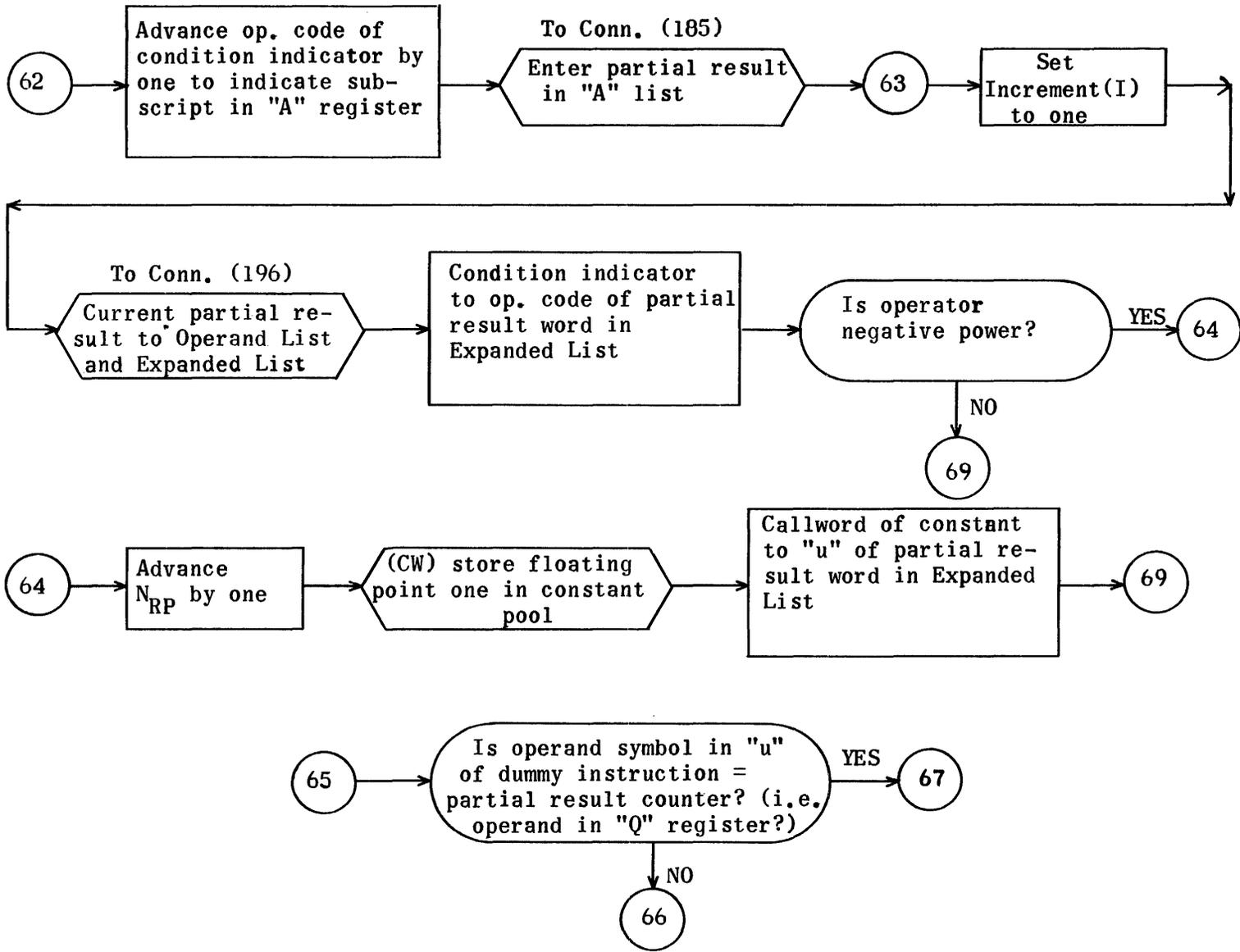


Equation Redundancy Check (Integral Power Operator)
Subroutine To Check for Redundancy of Integral Power Operator

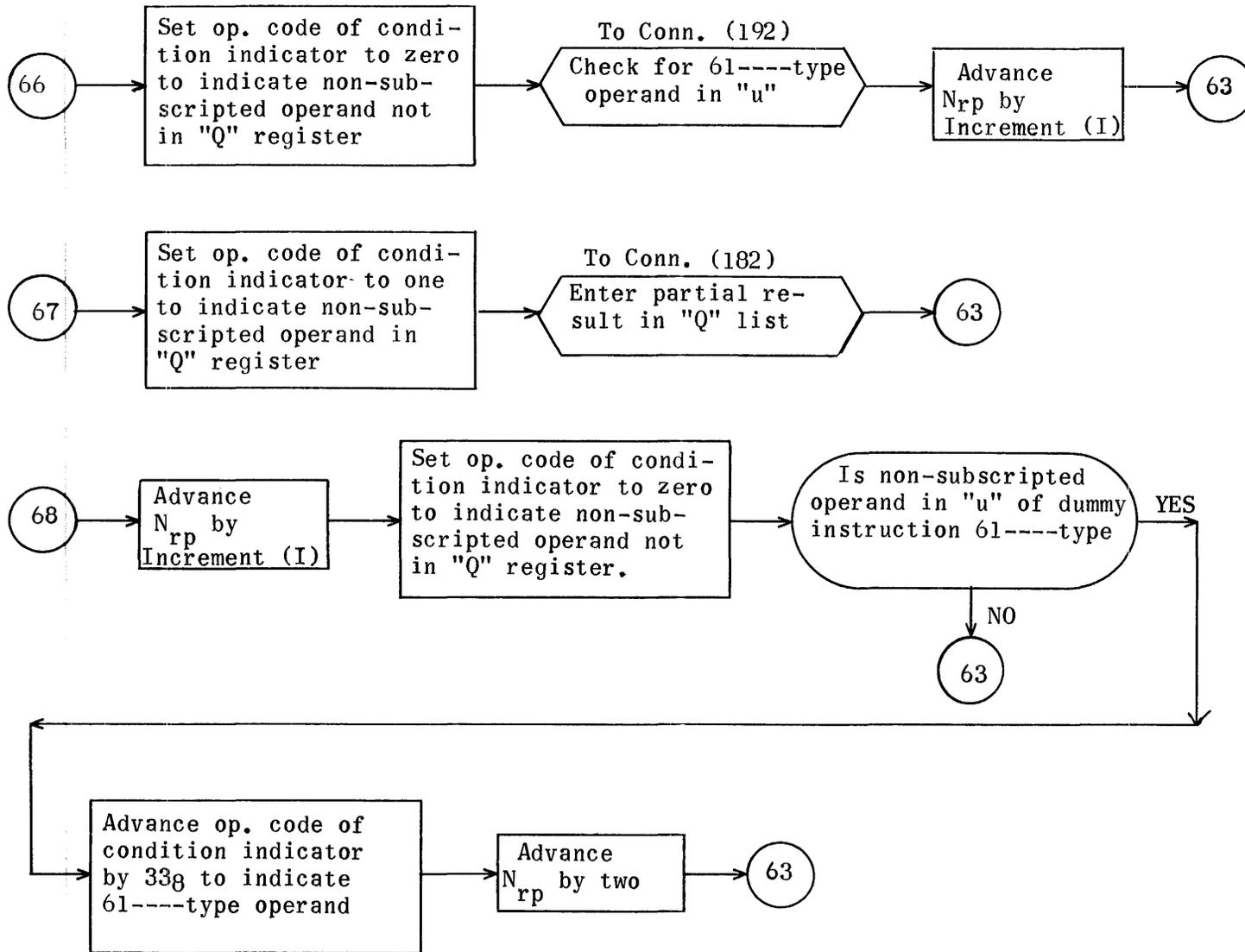


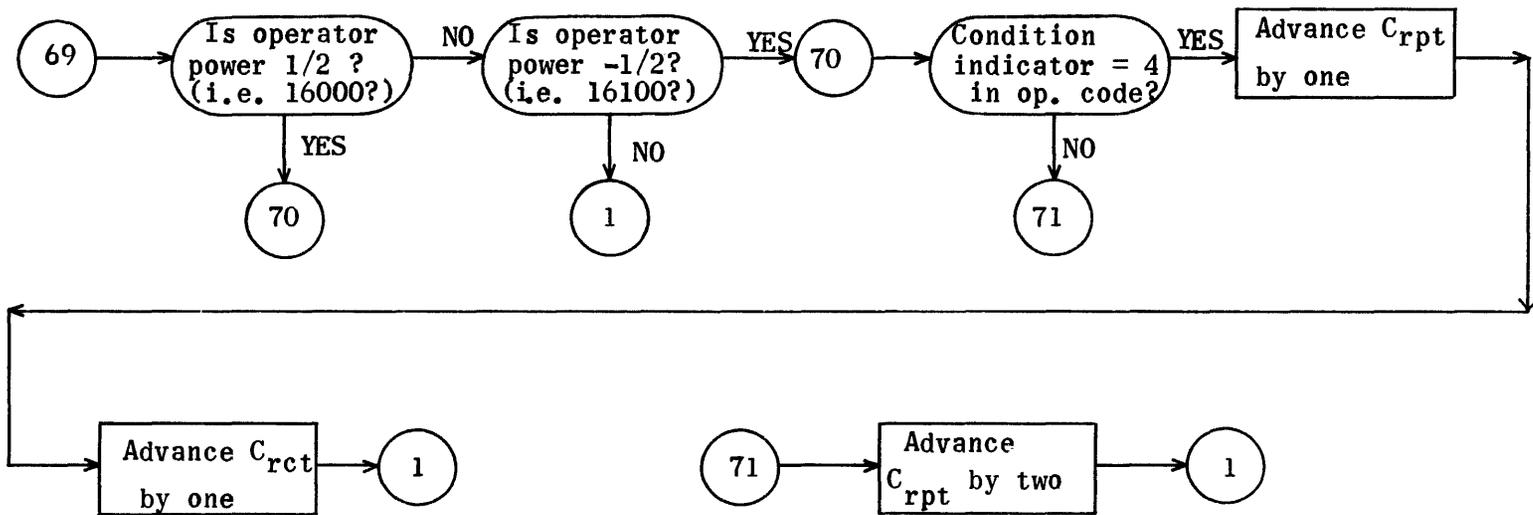
1248





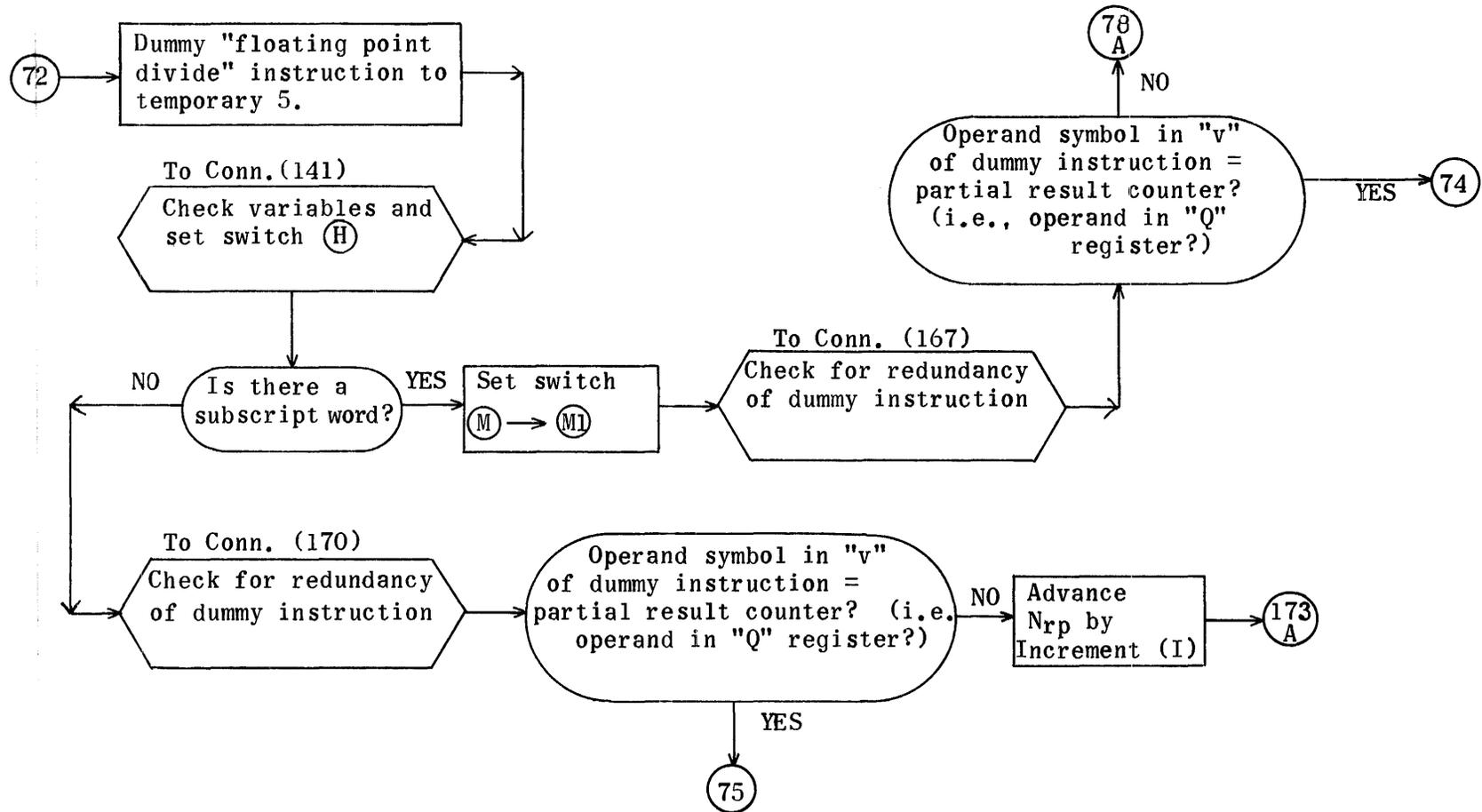
1250

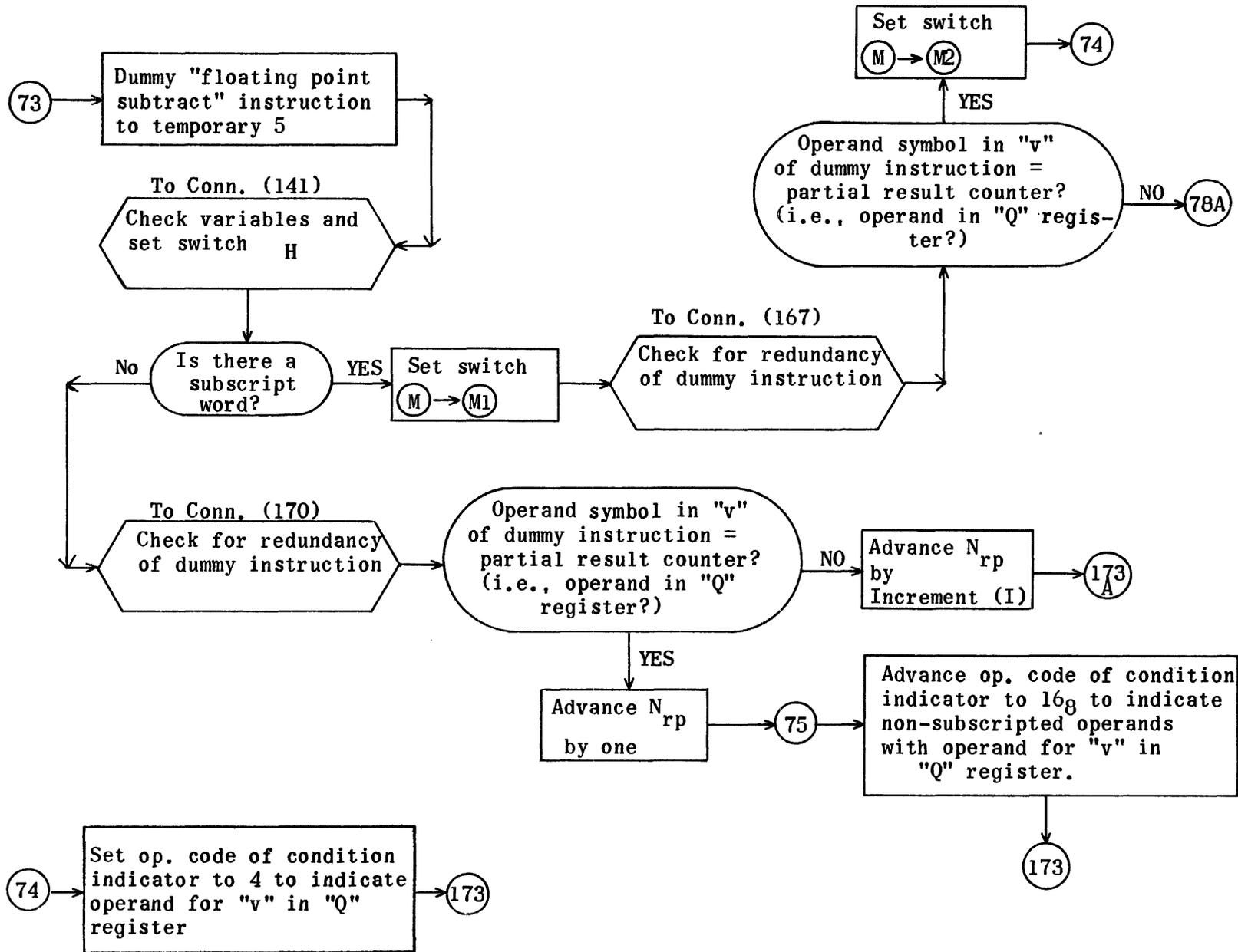


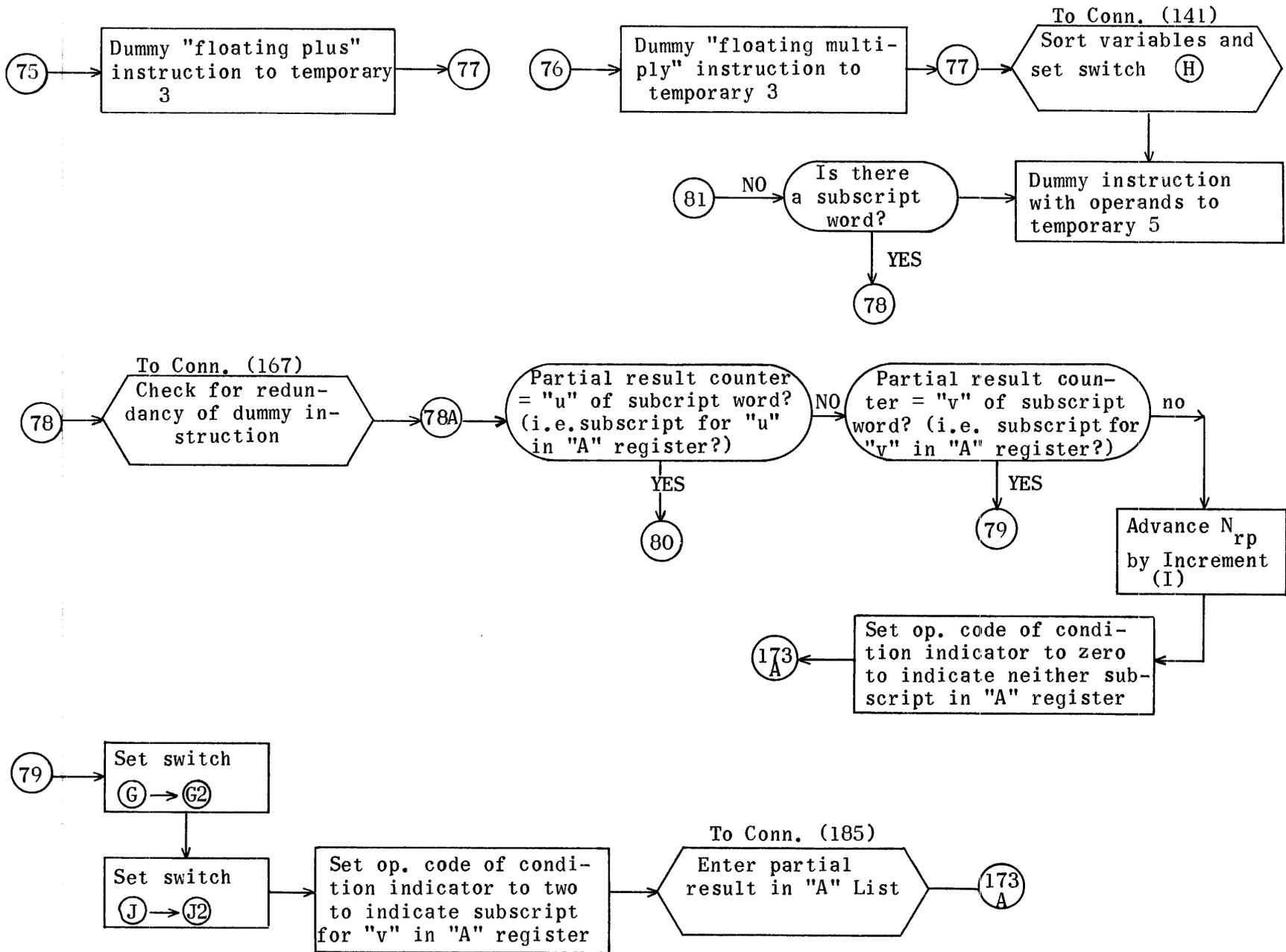


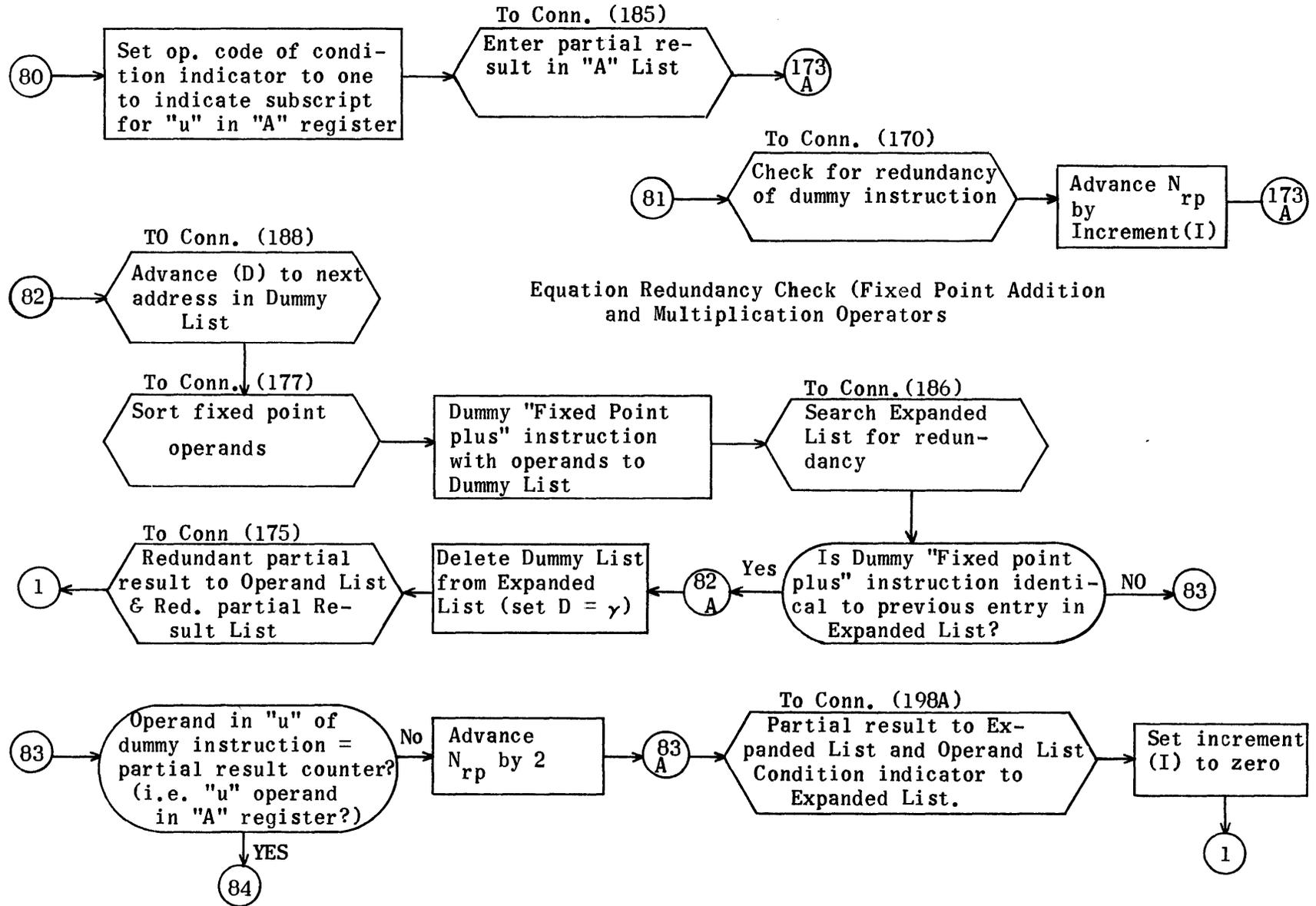
1252

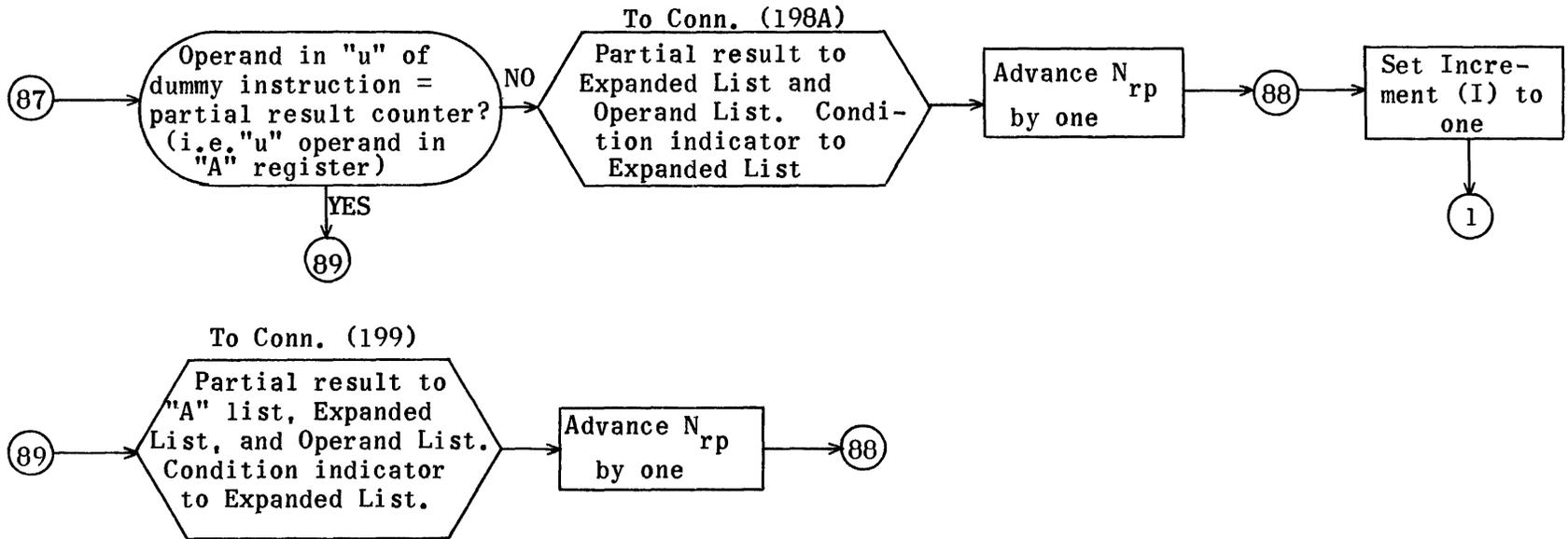
Equation Redundancy Check (Floating Point Divide and Subtract Operators)



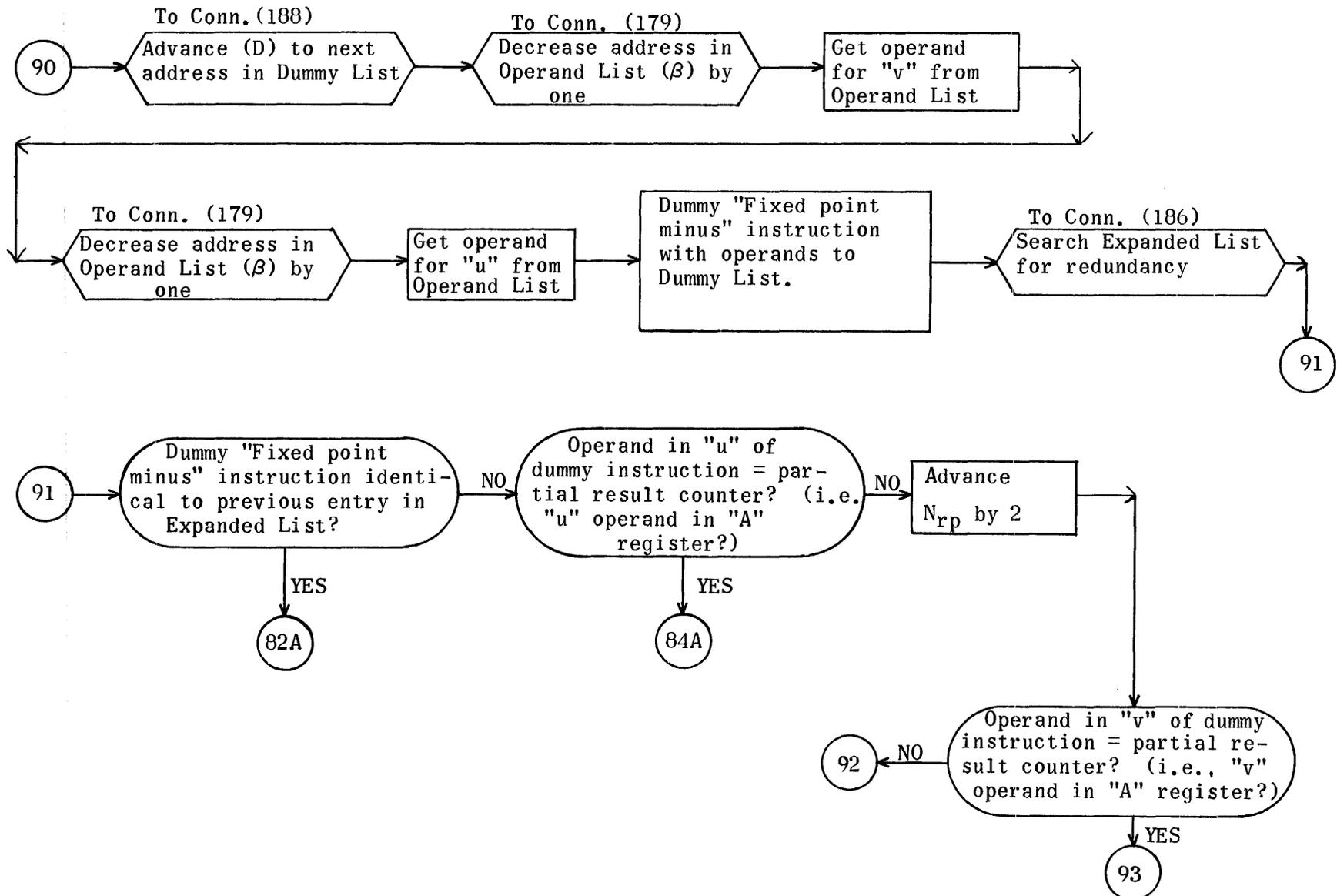


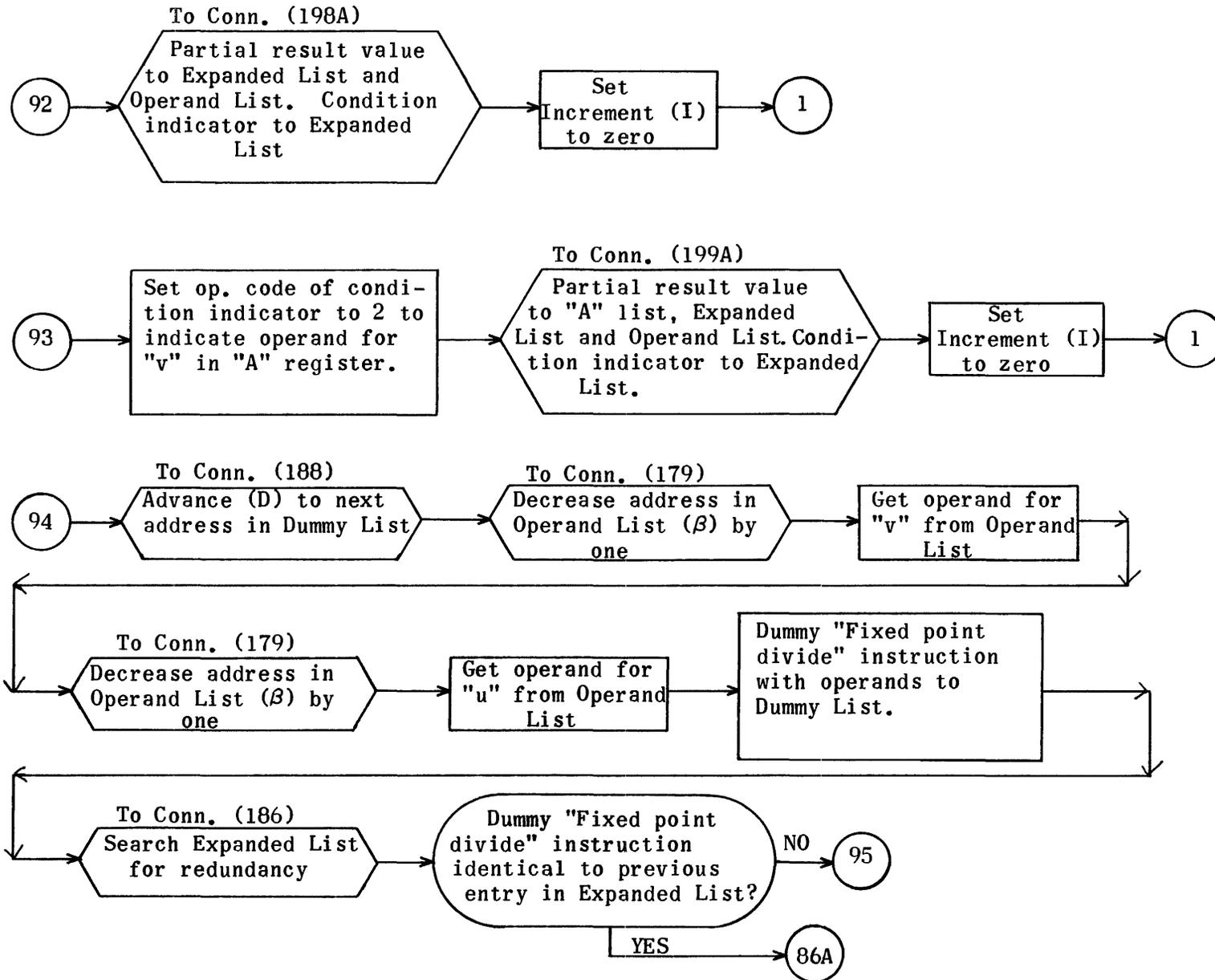


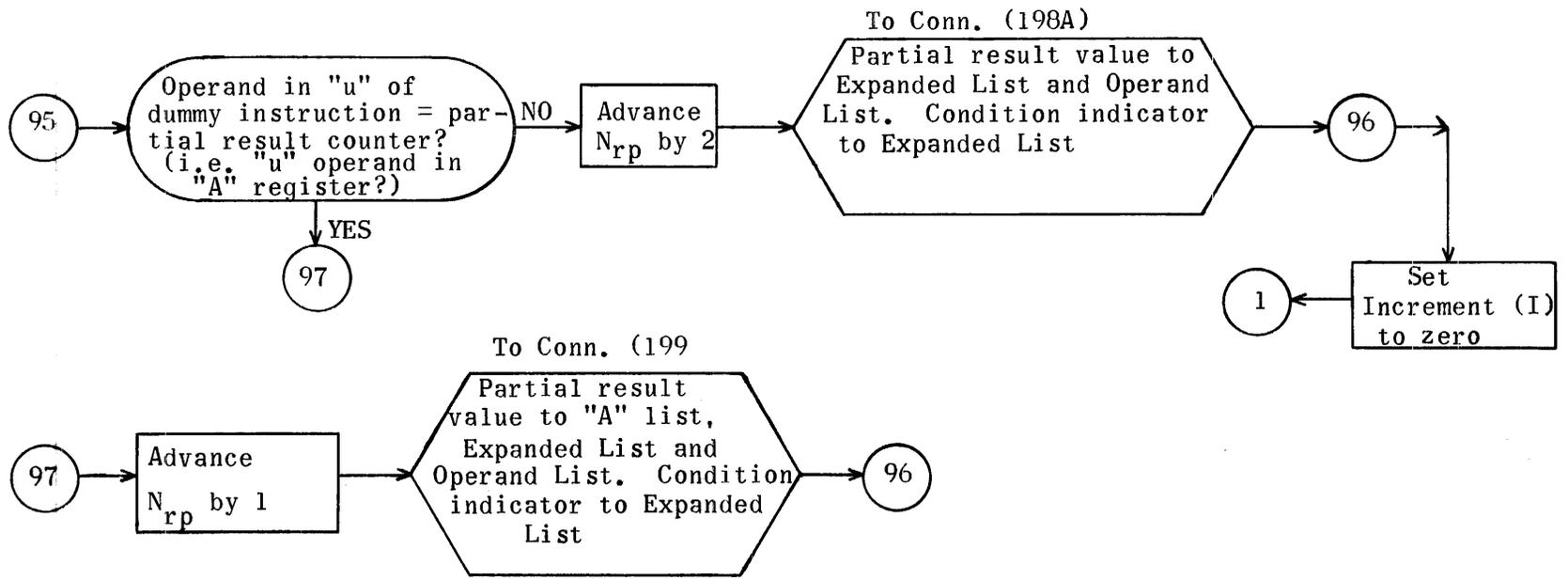




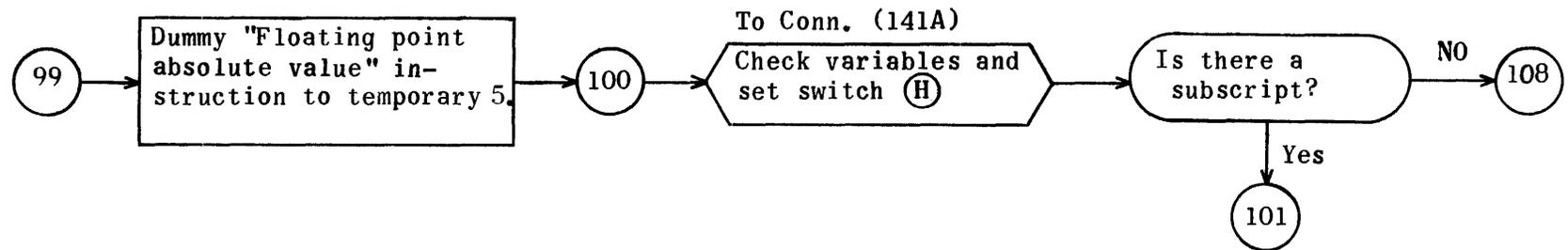
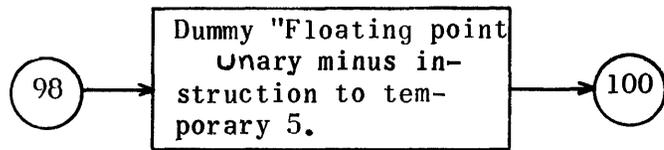
Equation Redundancy Check (Fixed Point Subtraction and Division Operators)

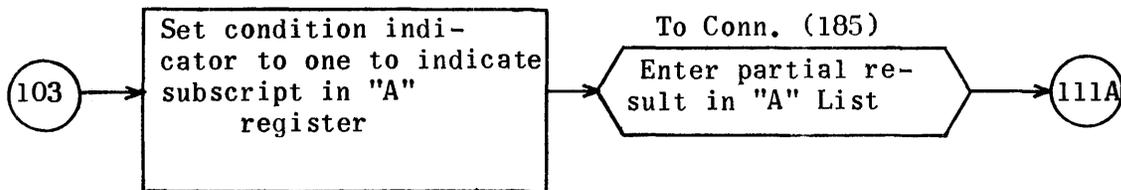
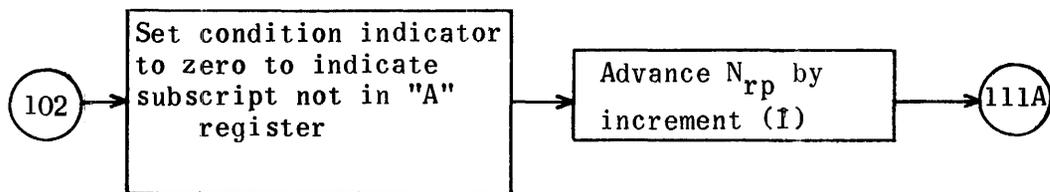
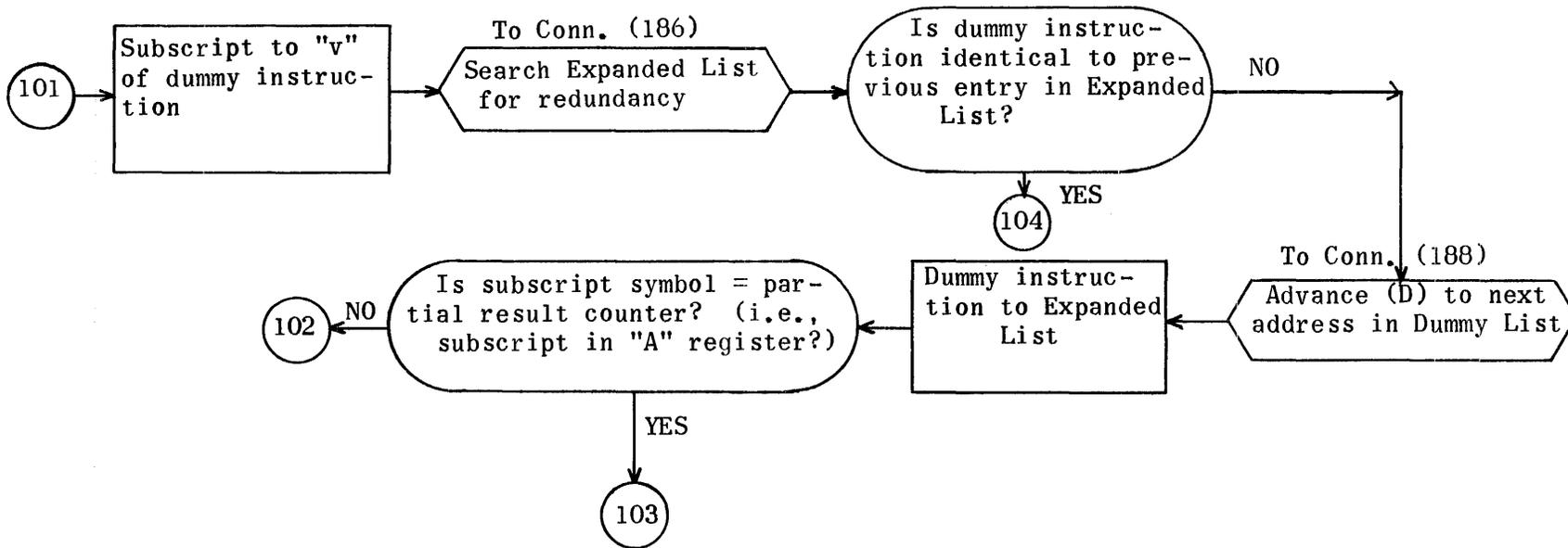


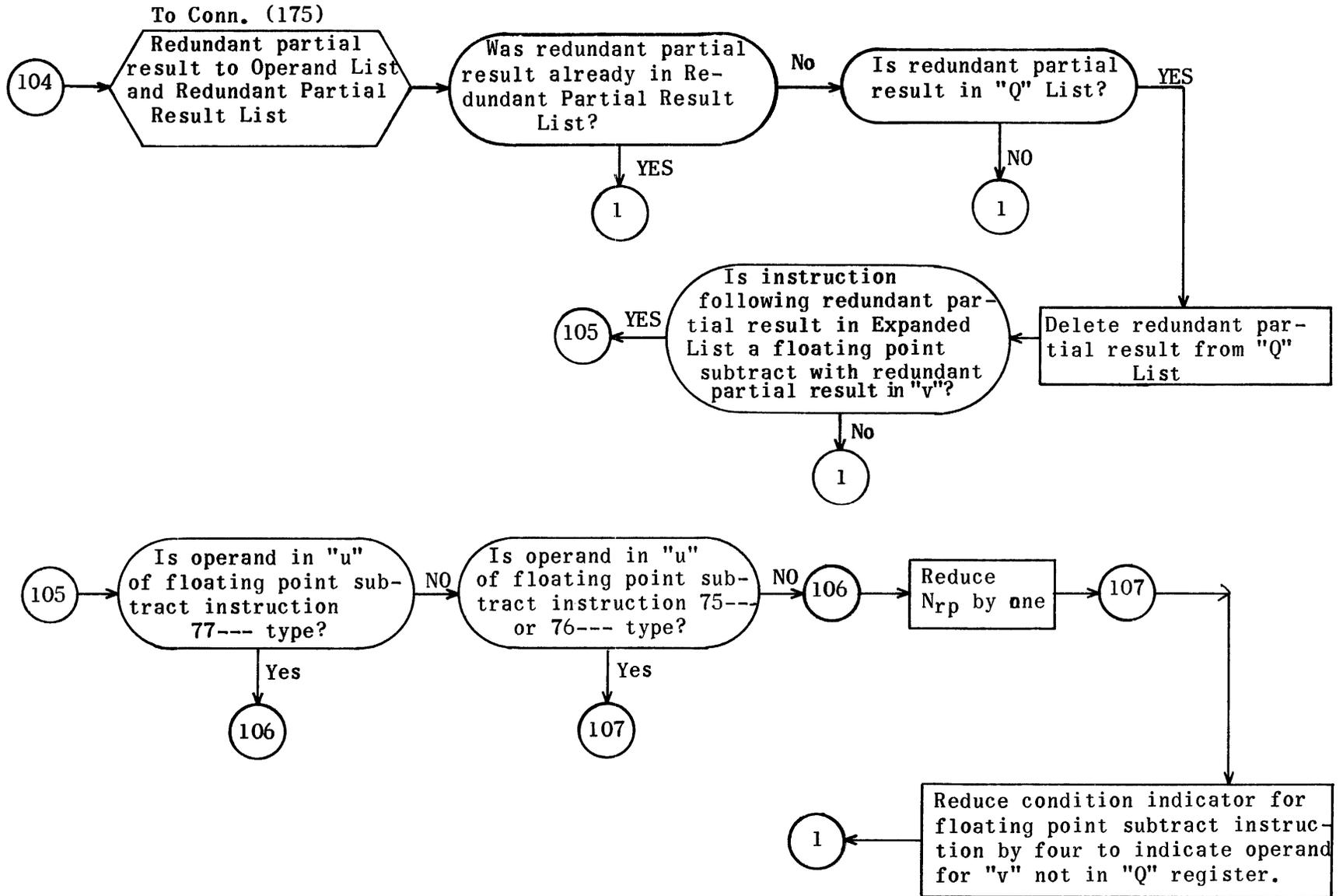


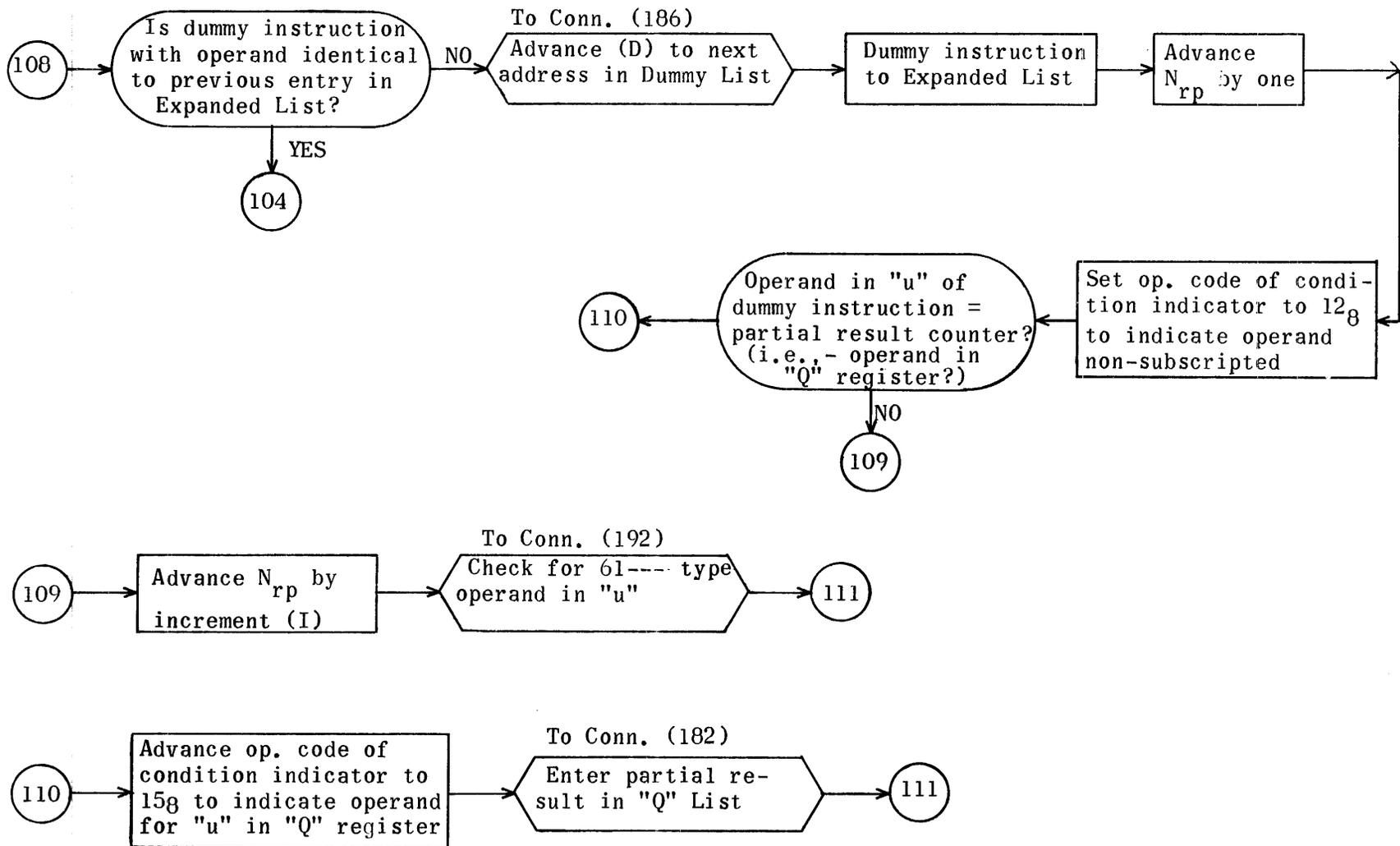


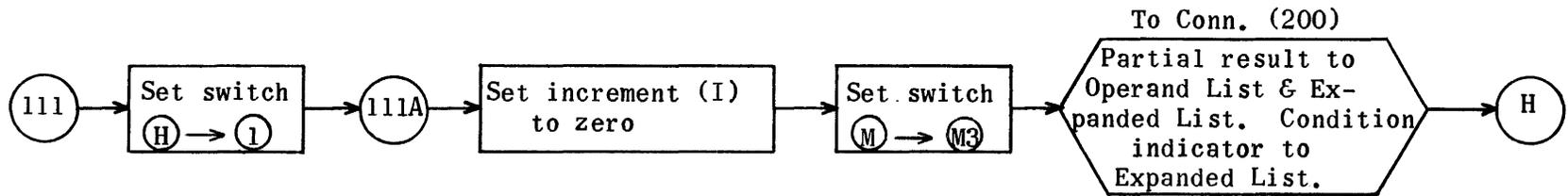
Equation Redundancy Check (Floating Point Unary Minus and Absolute Value Operators)



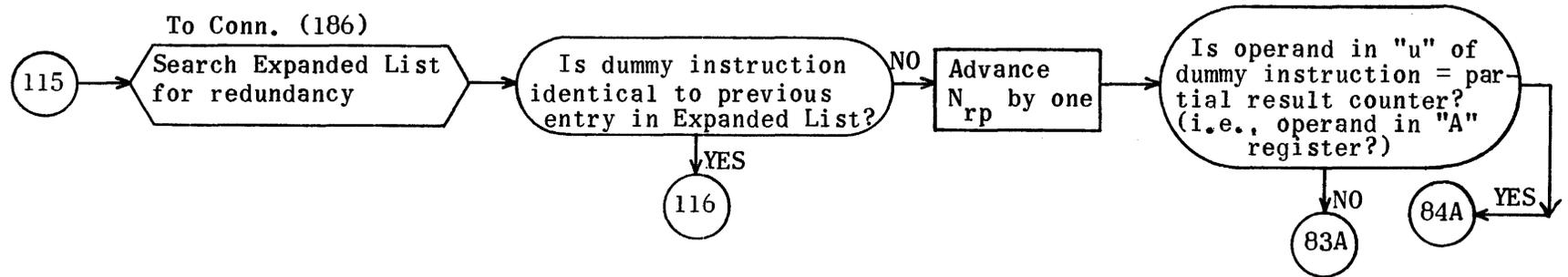
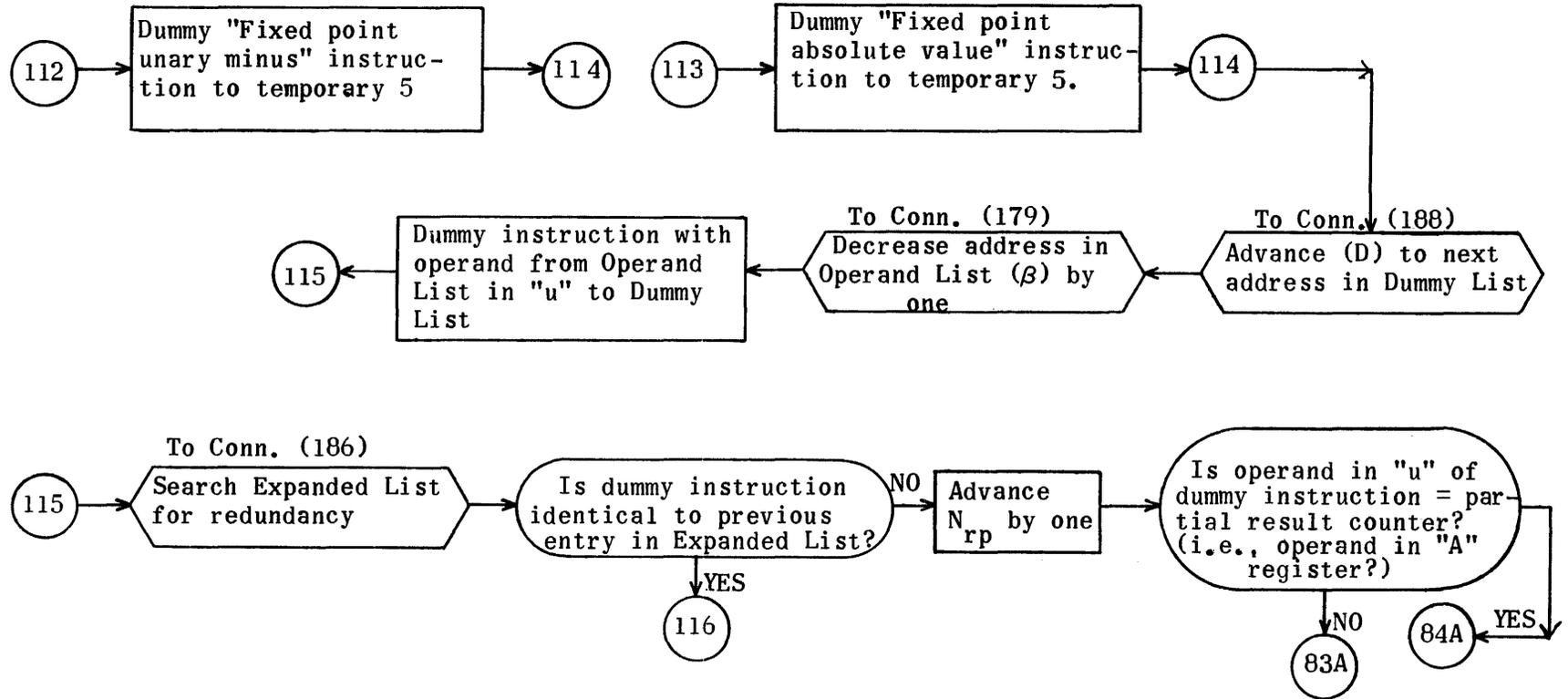


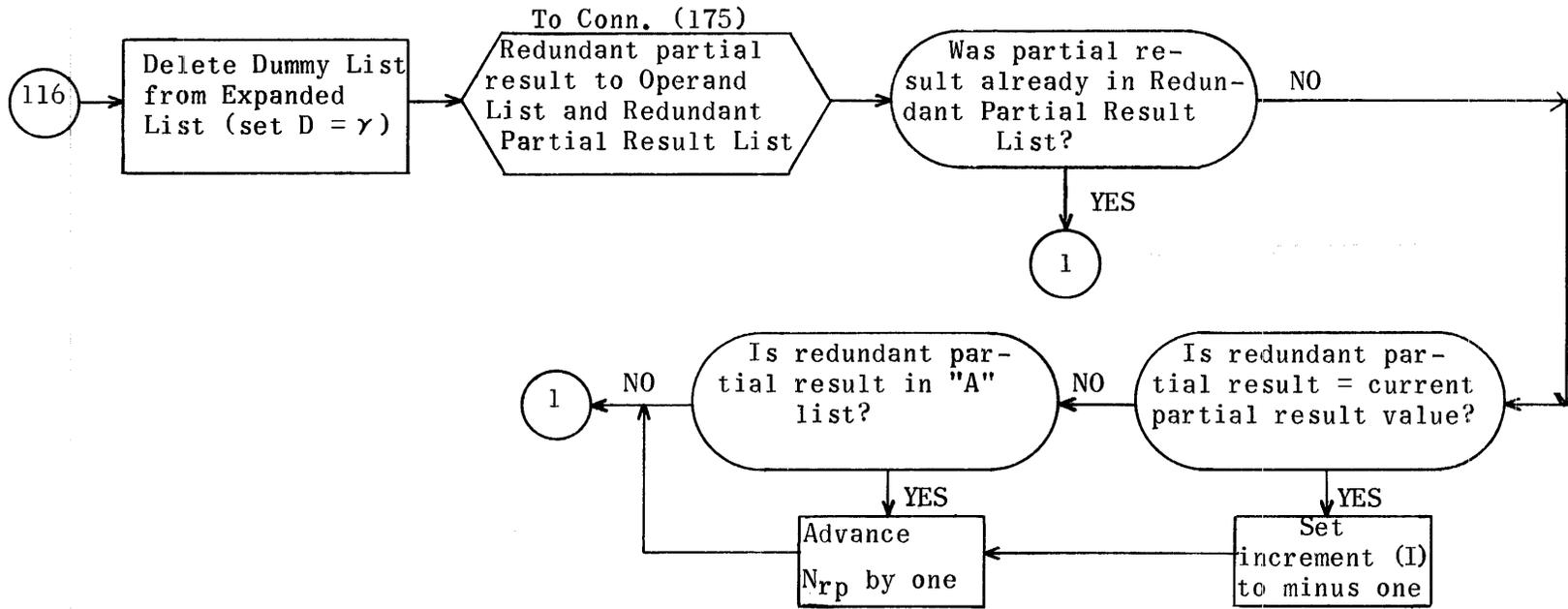




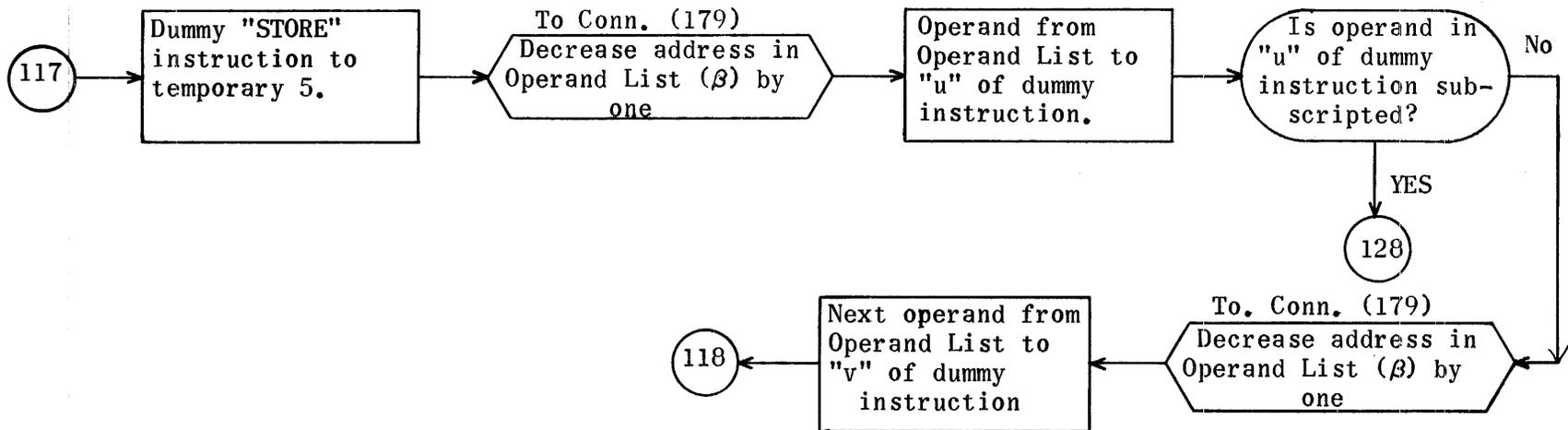


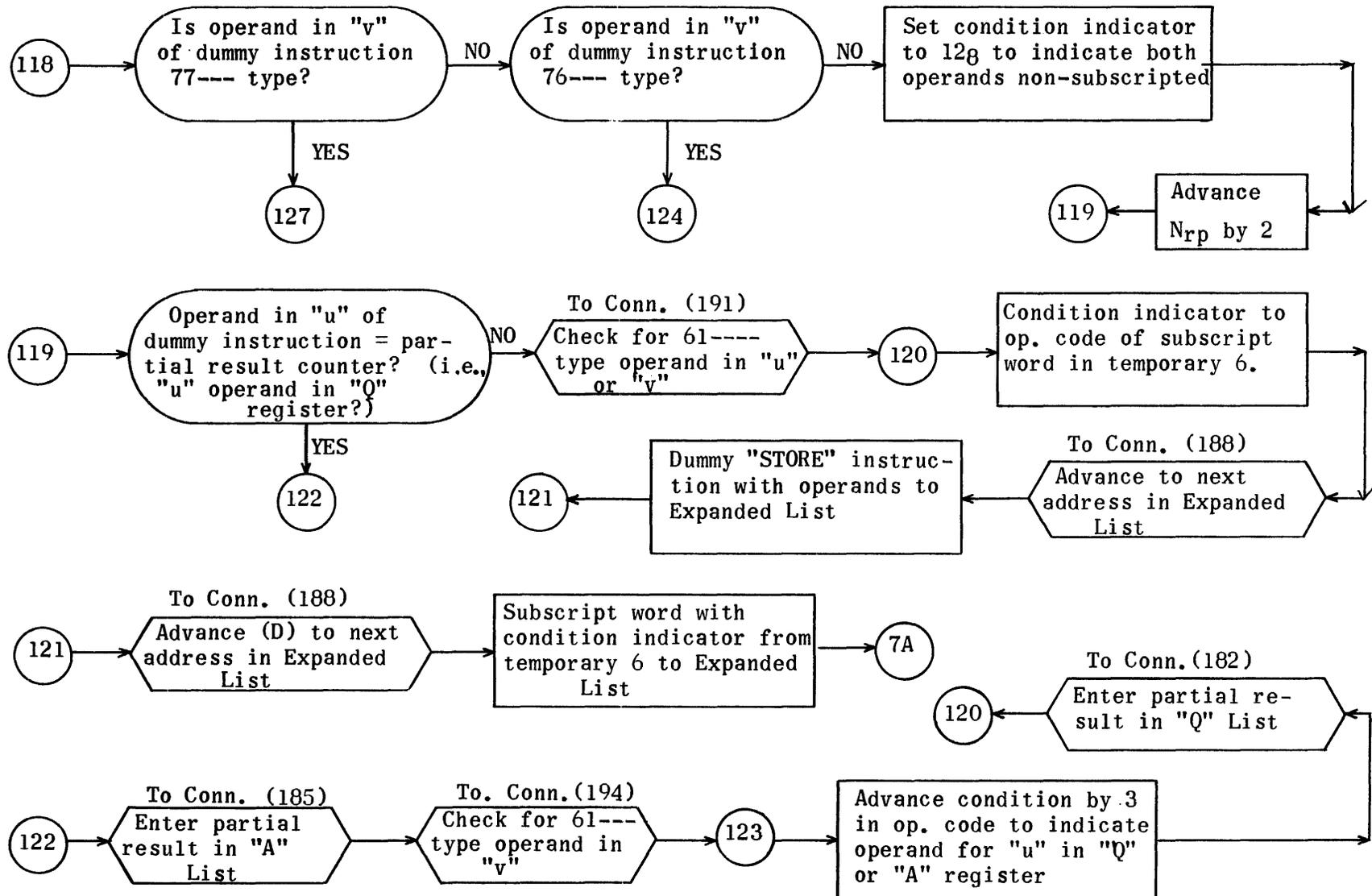
Equation Redundancy Check (Fixed Point Unary Minus and Absolute Value Operators)



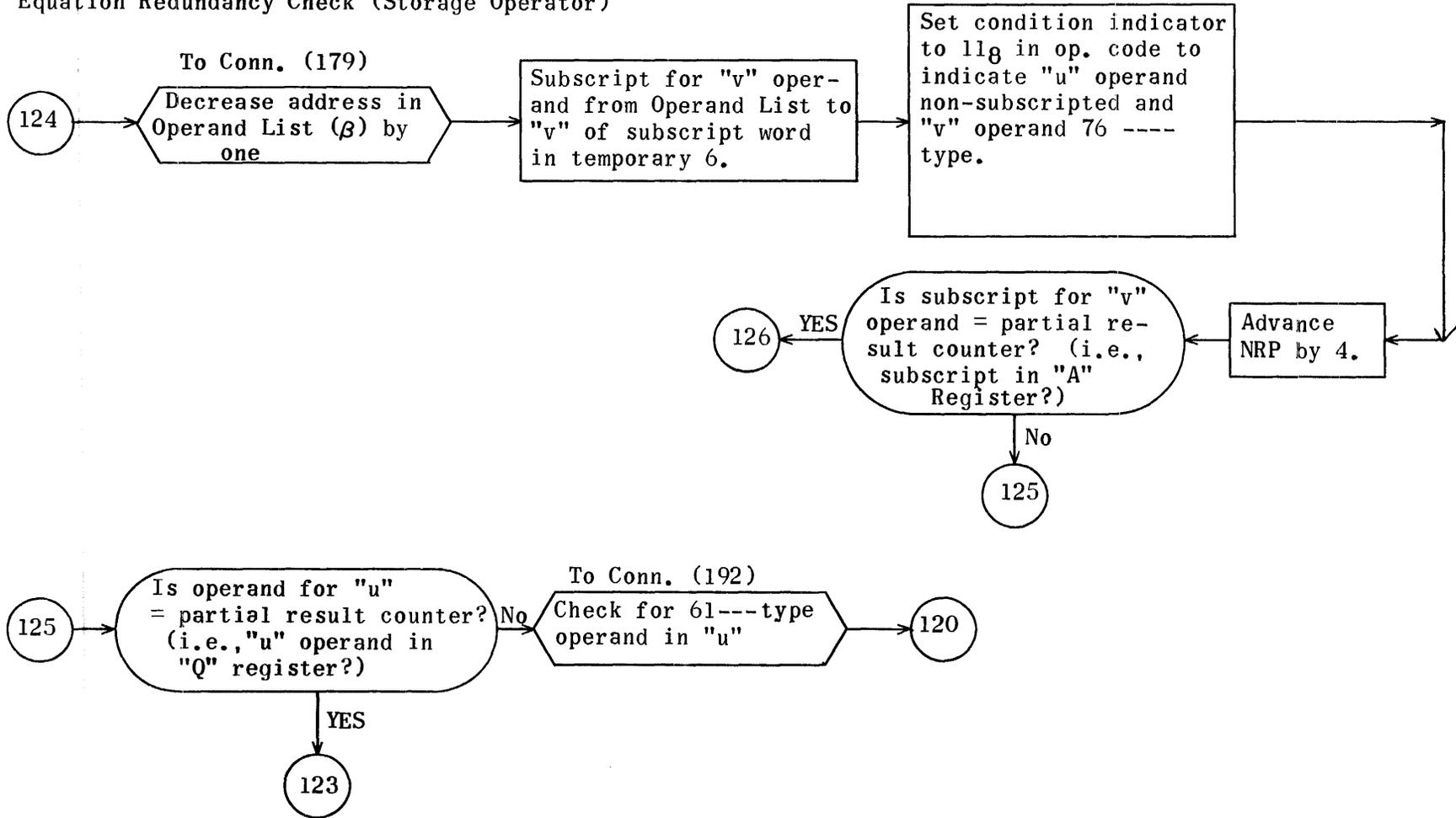


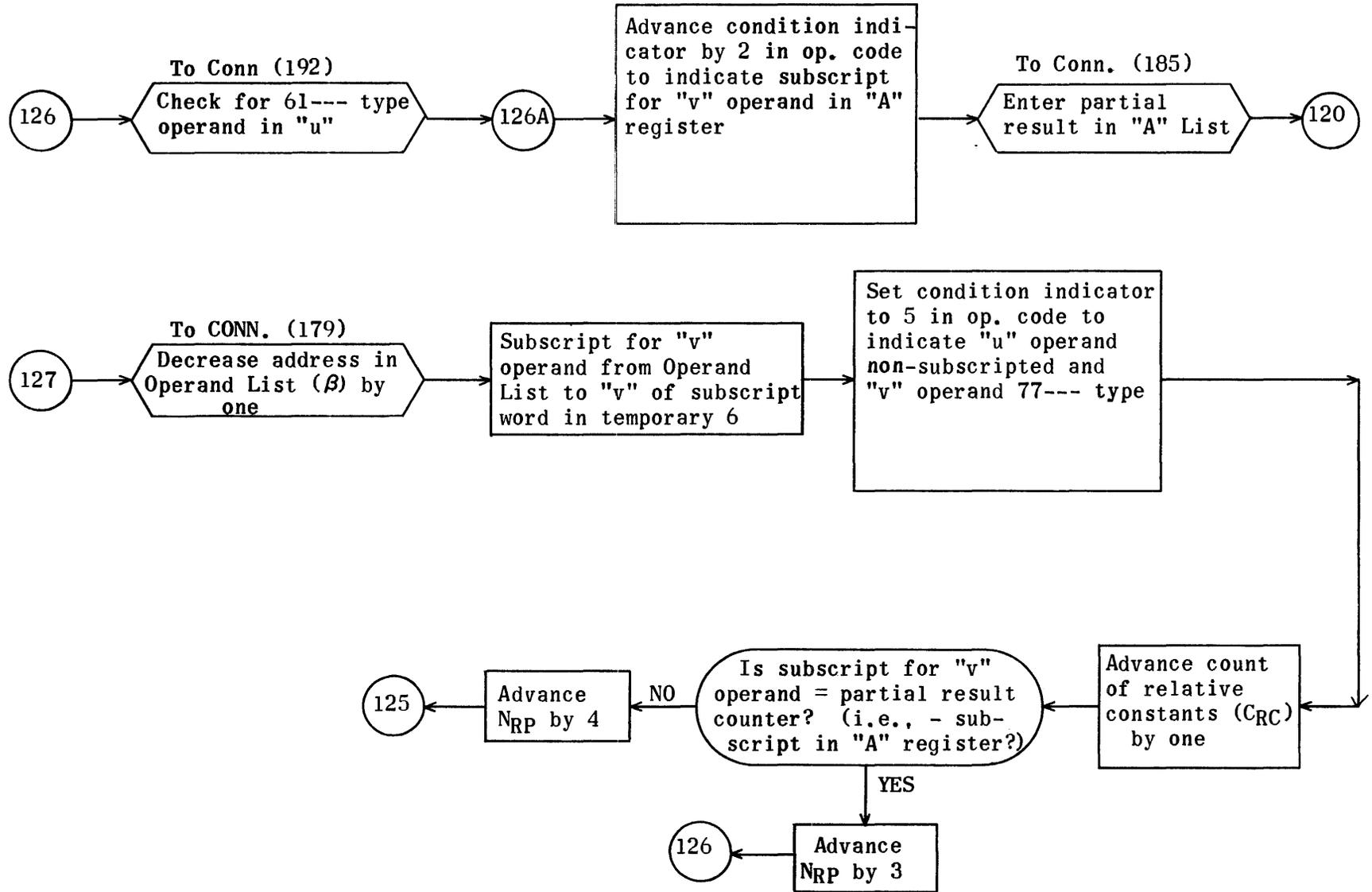
Equation Redundancy Check (Storage Operator)

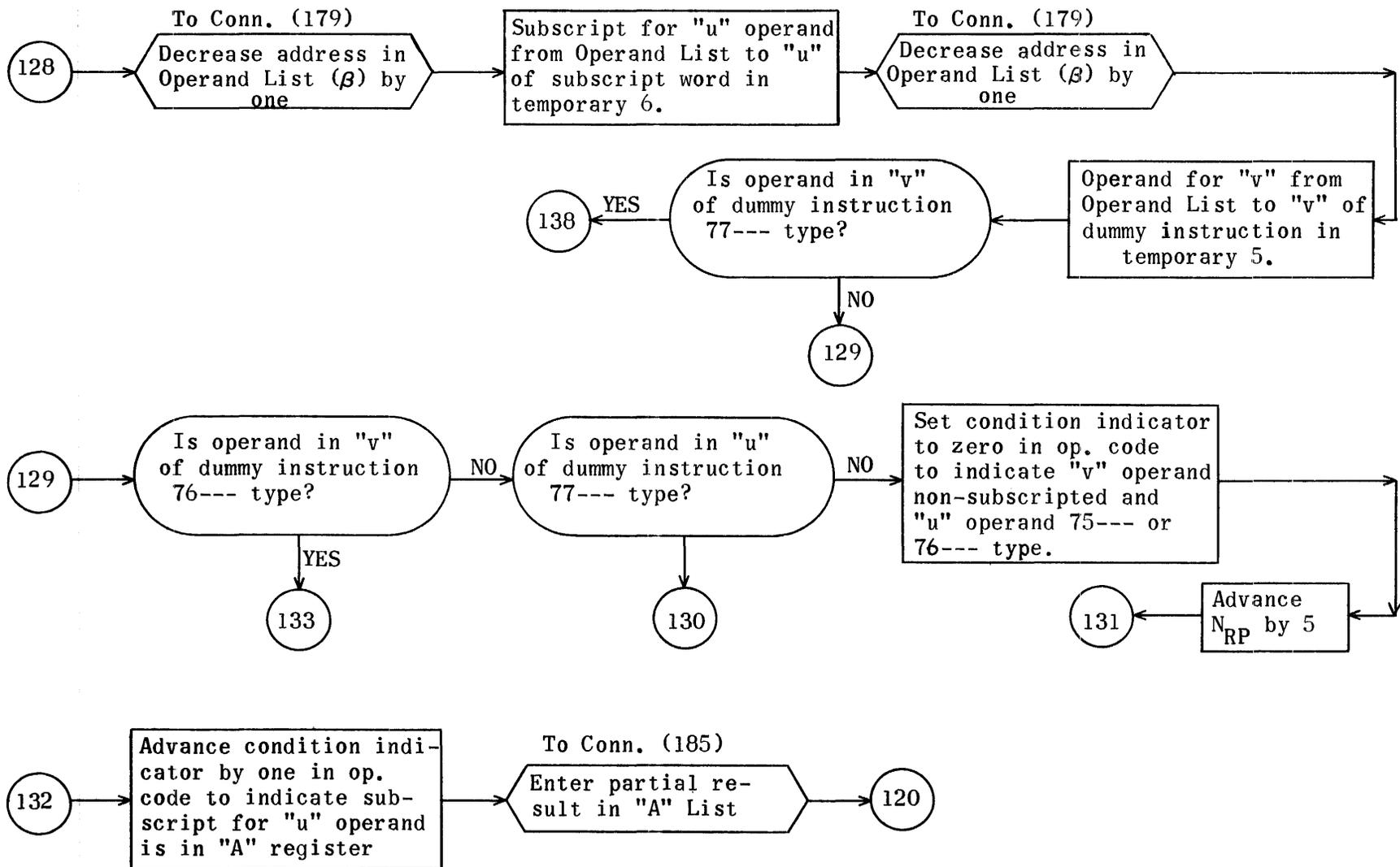


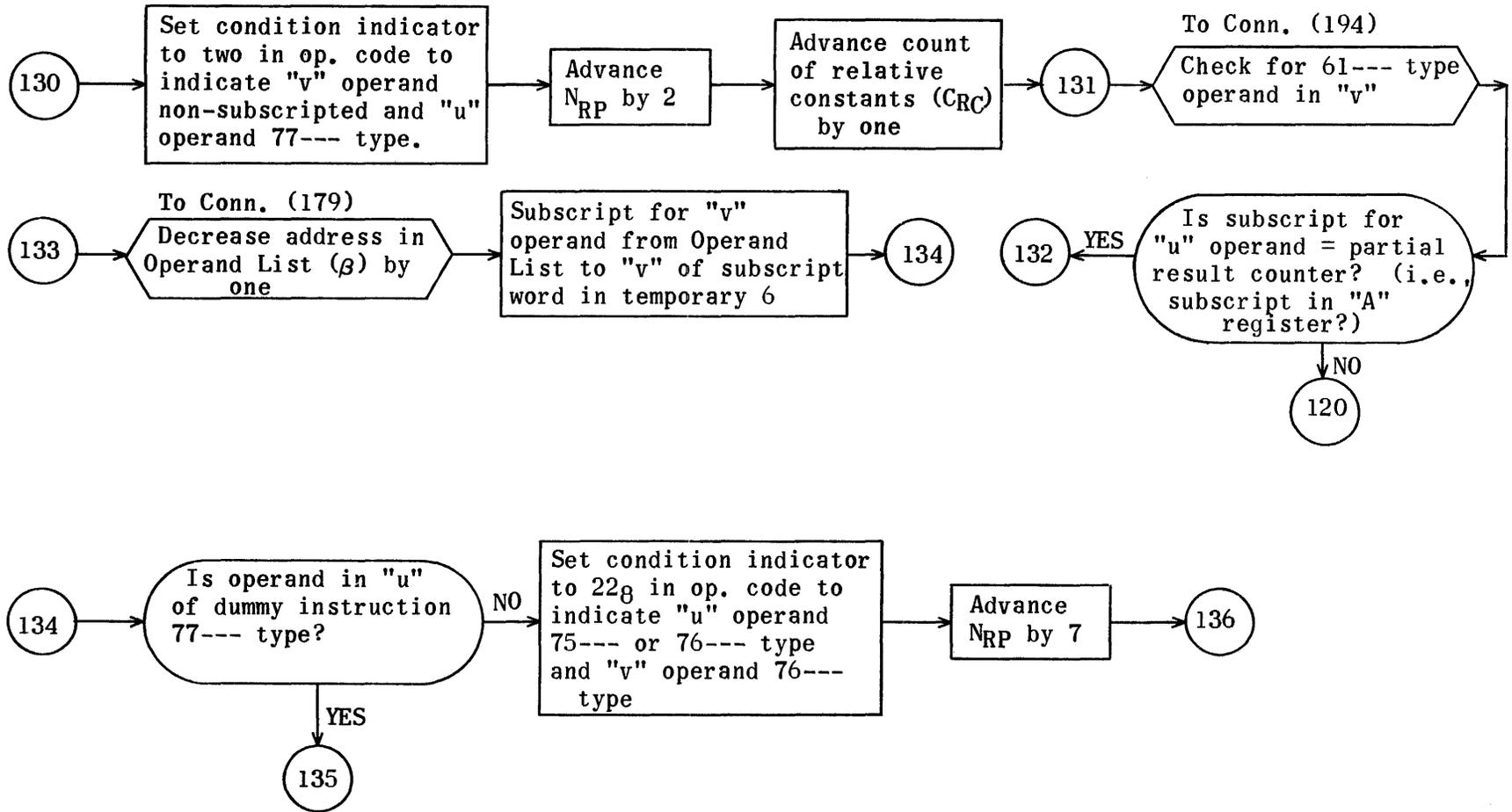


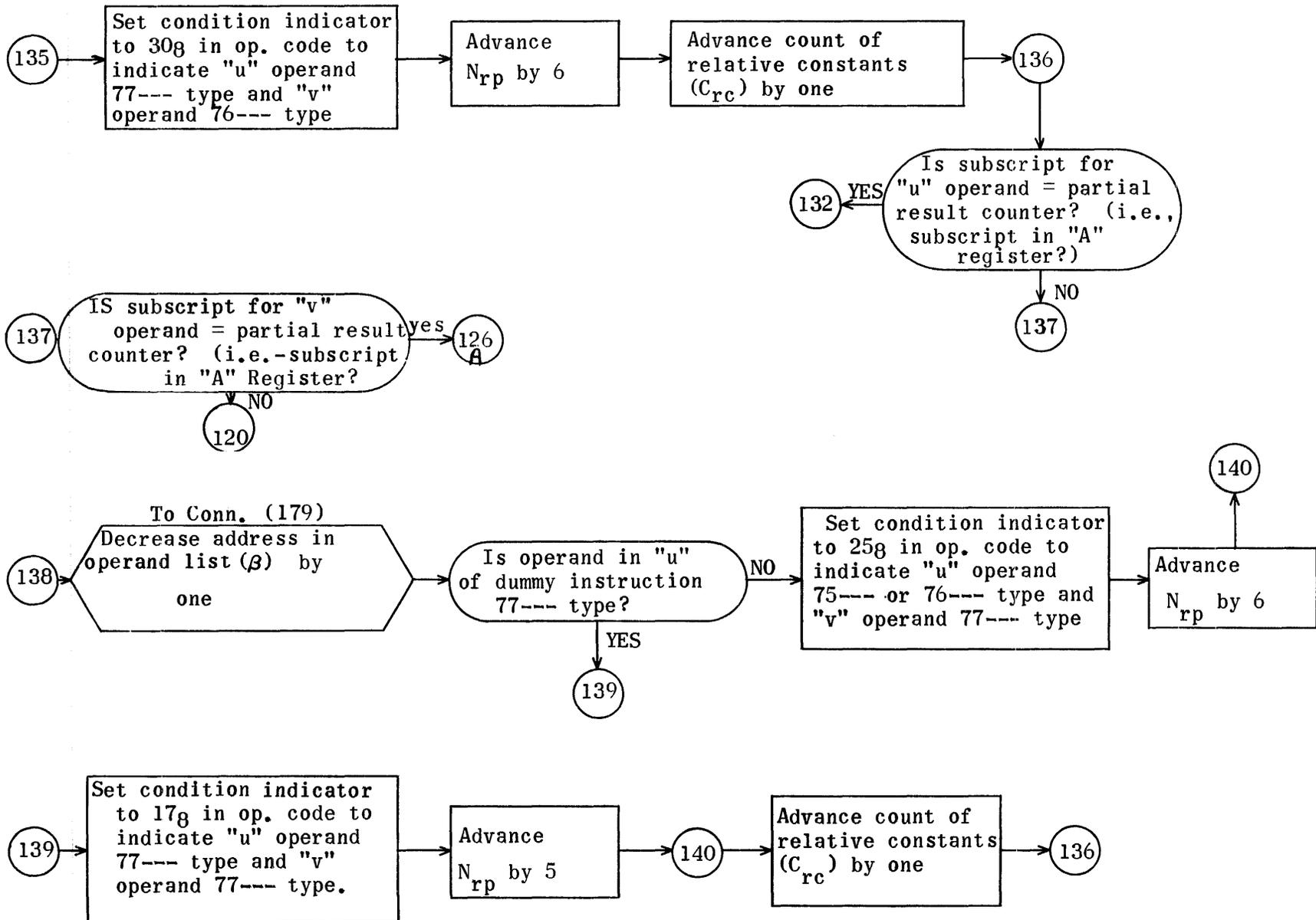
Equation Redundancy Check (Storage Operator)



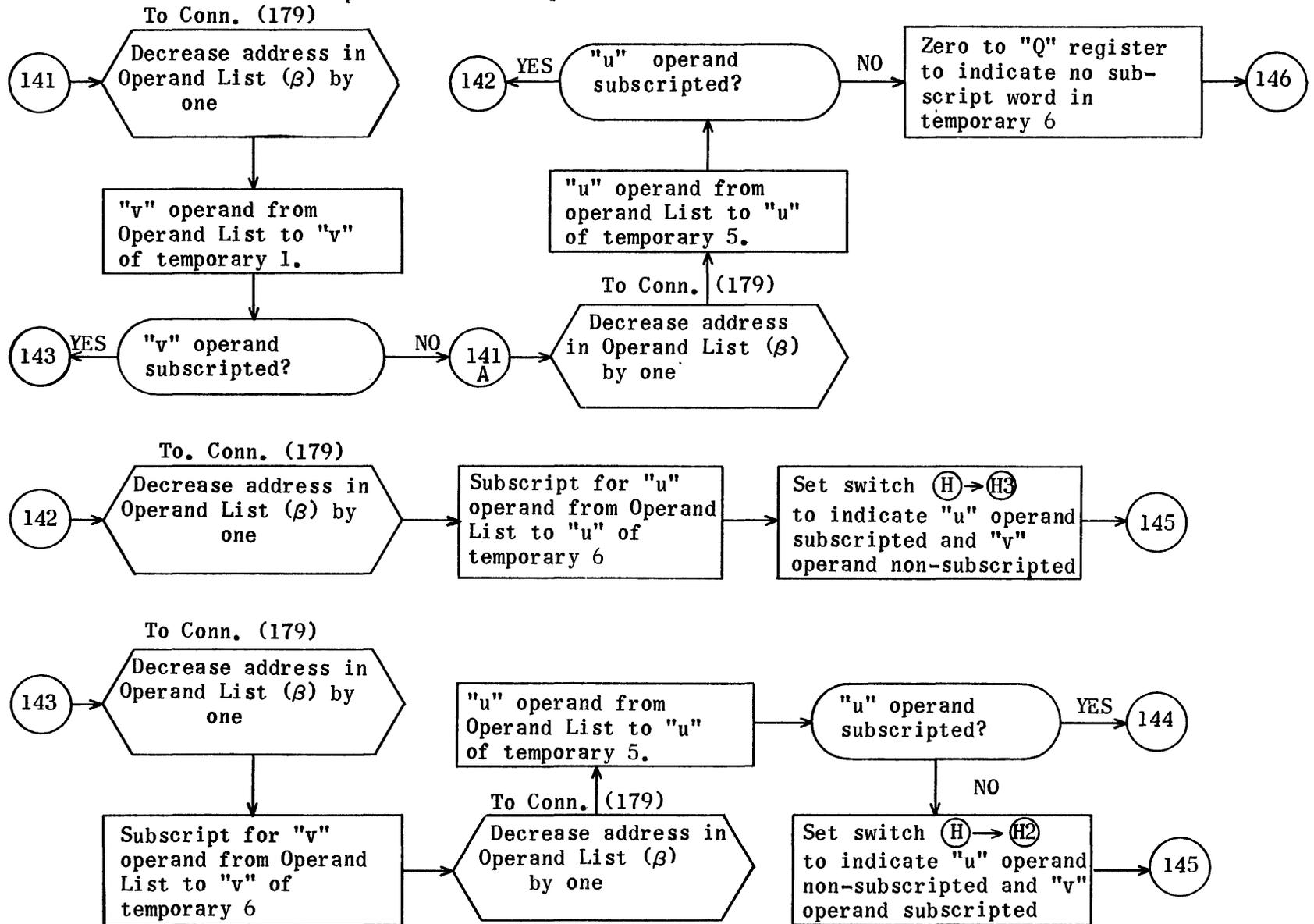


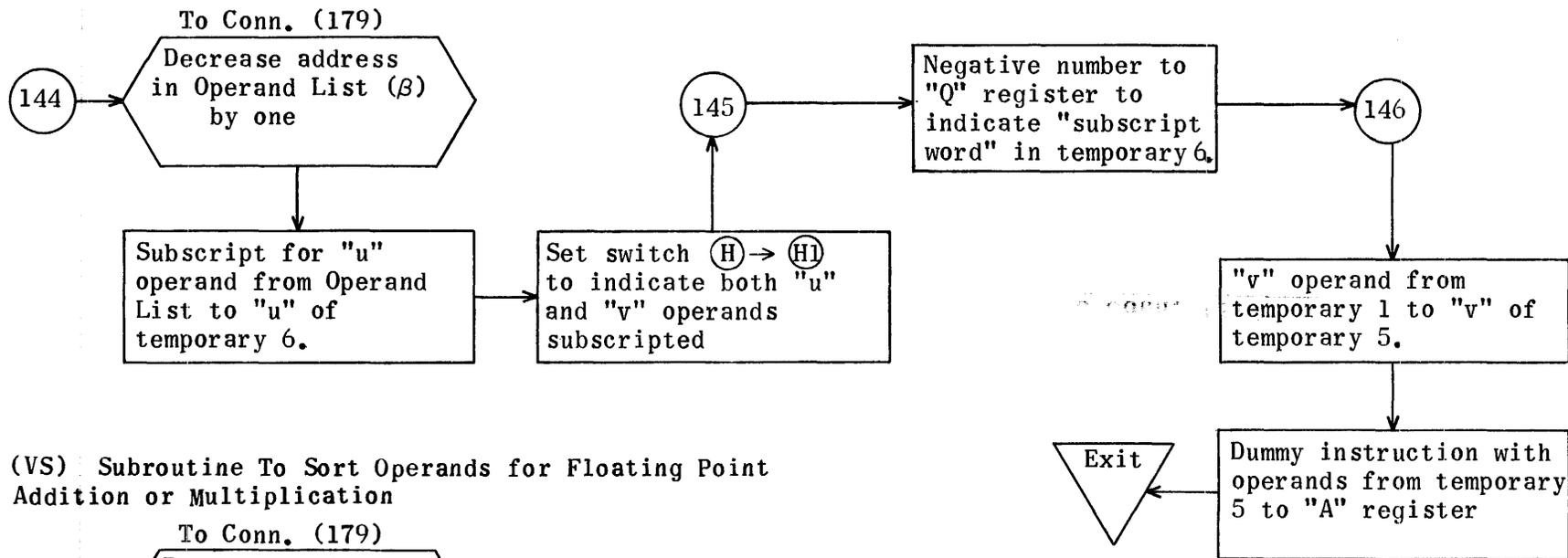




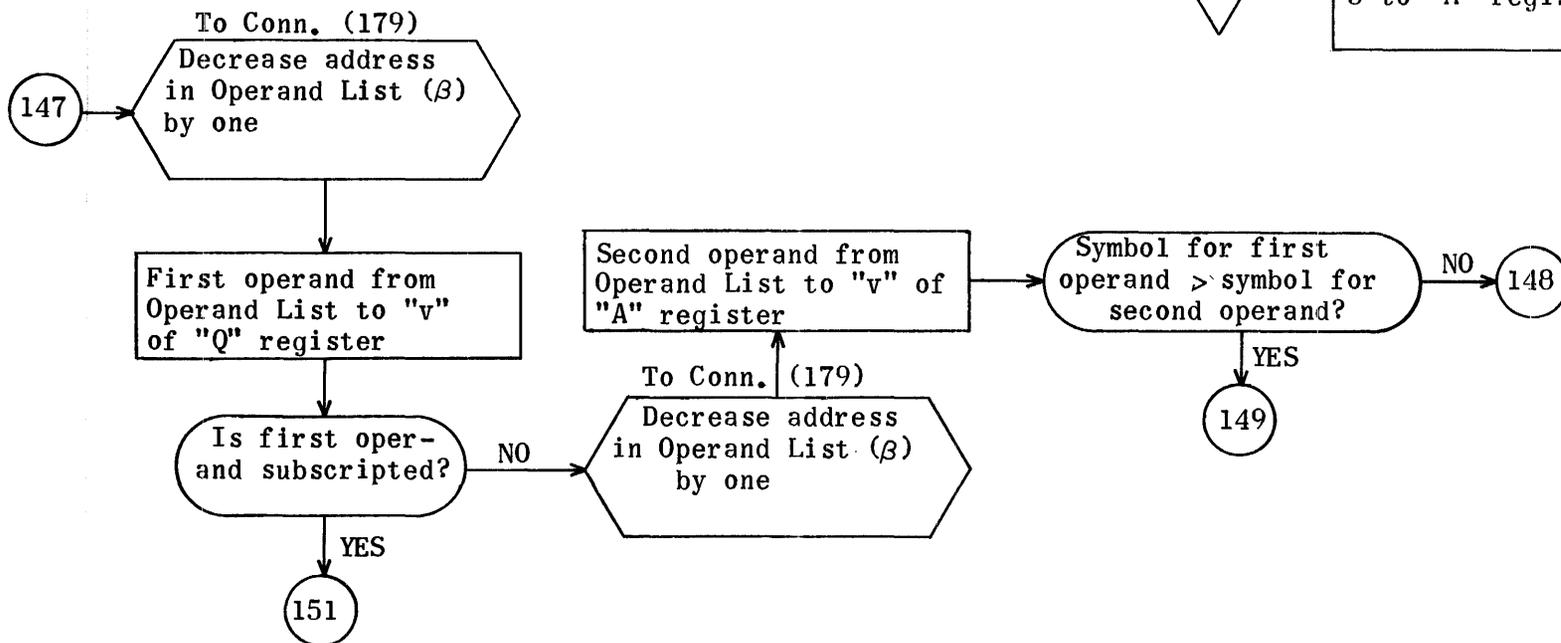


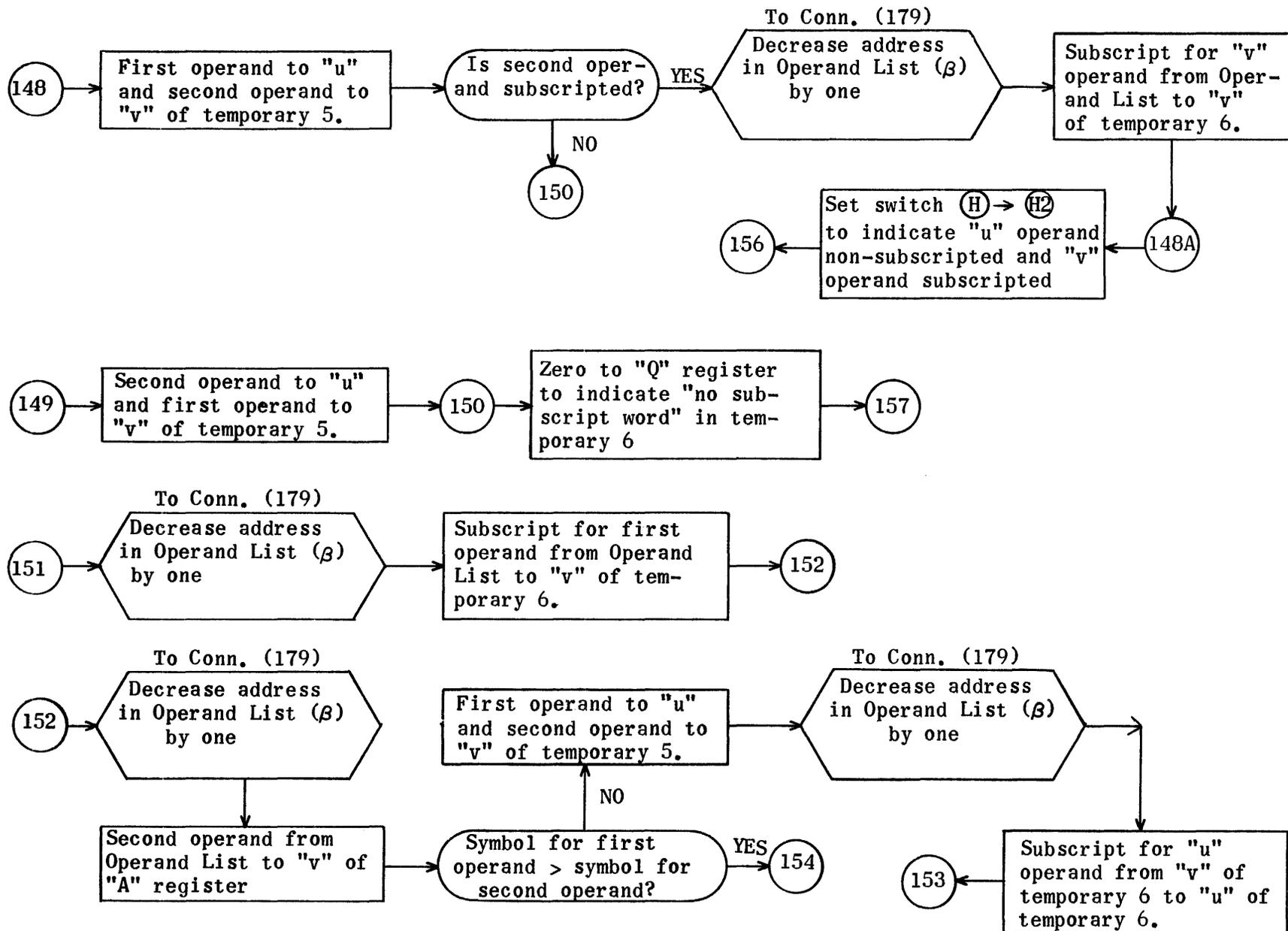
Equation Redundancy Check (Subroutine to Check Variables)

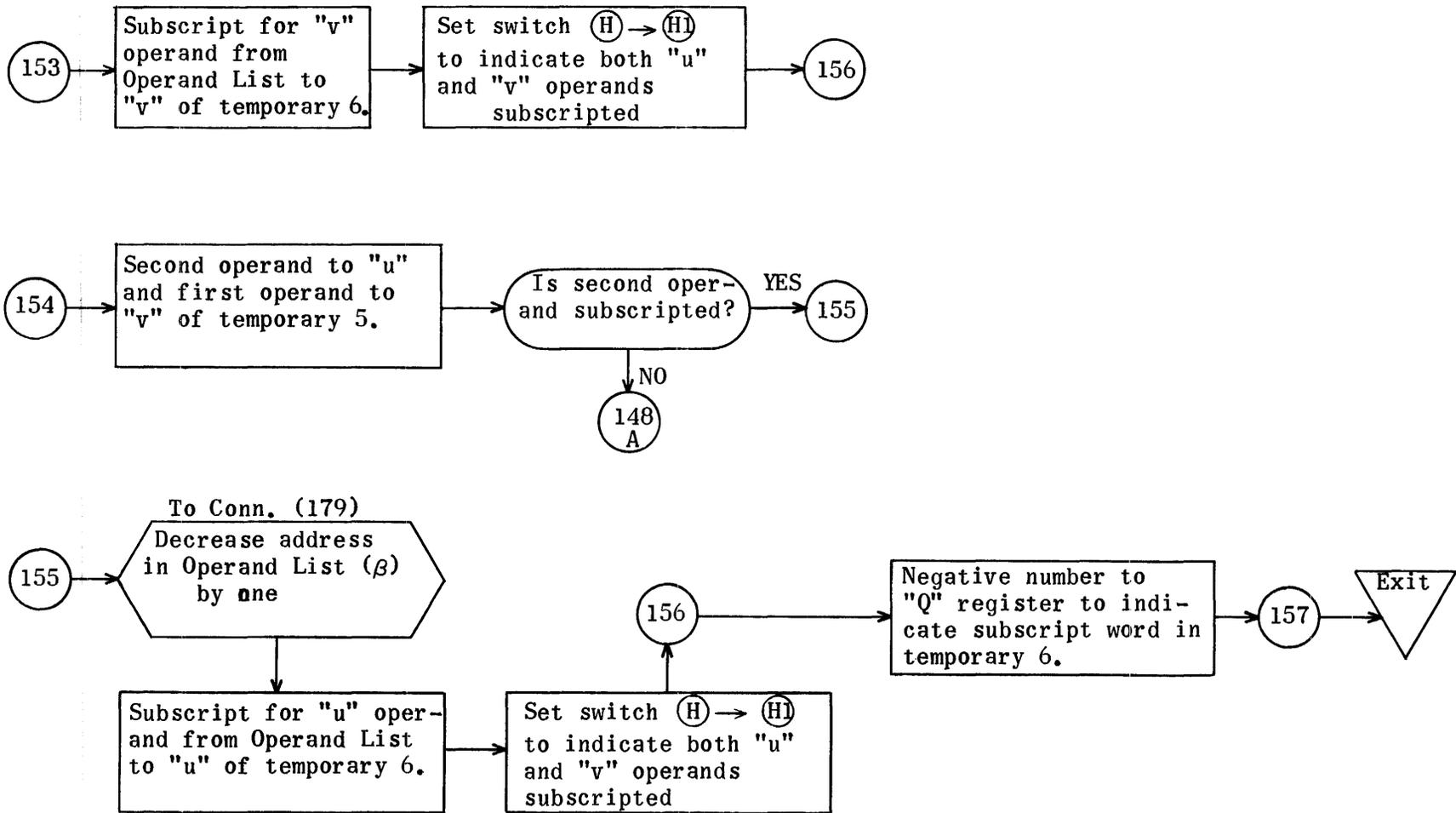




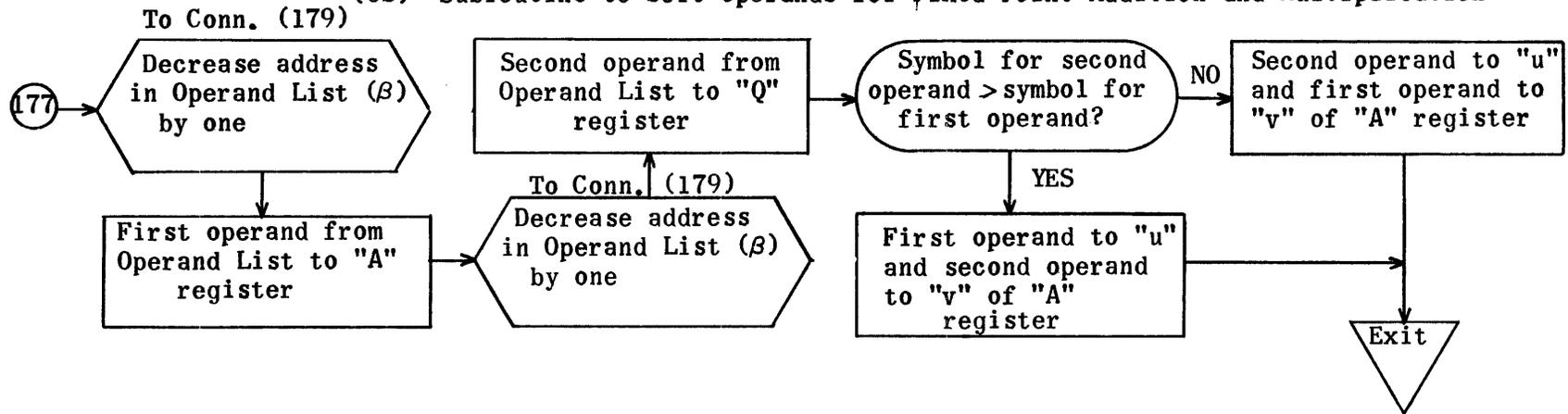
(VS) Subroutine To Sort Operands for Floating Point Addition or Multiplication



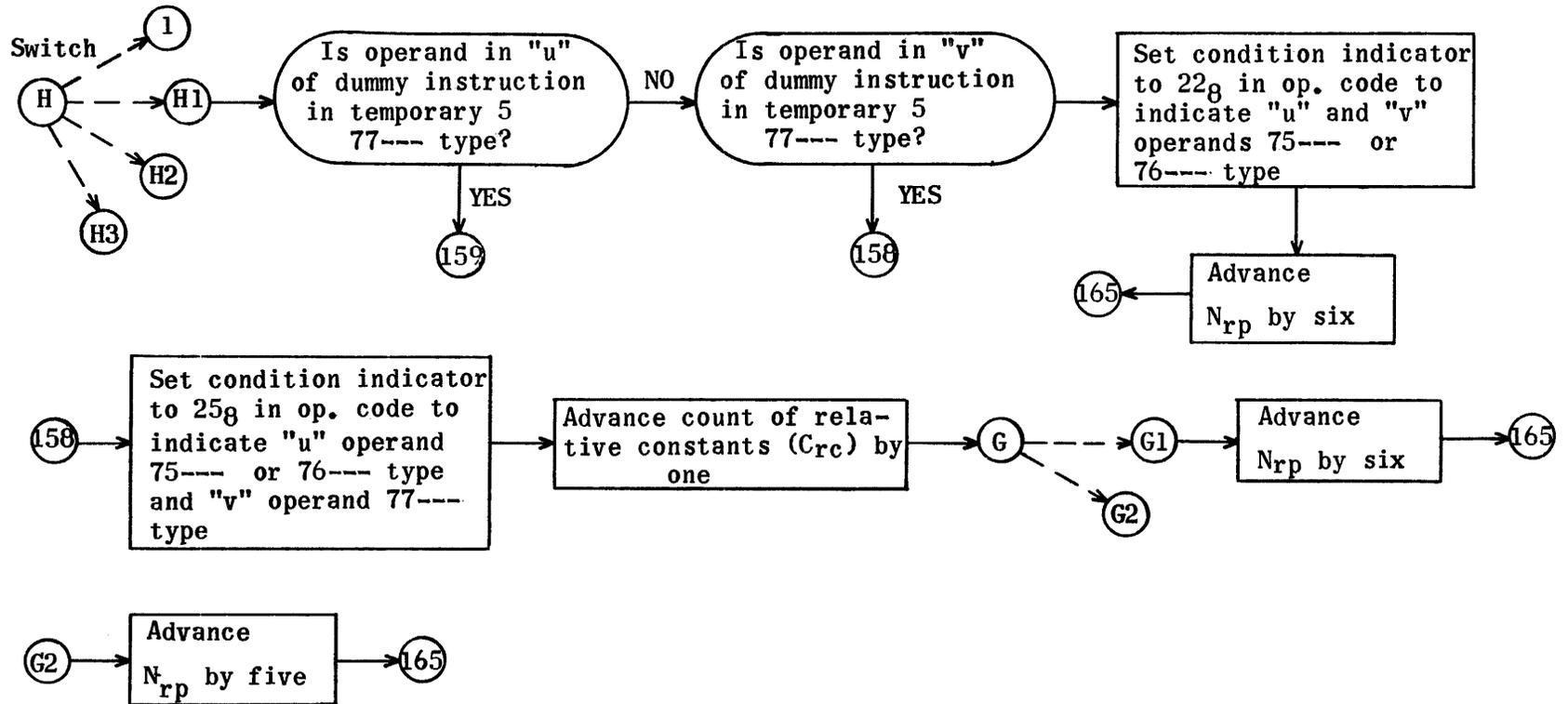


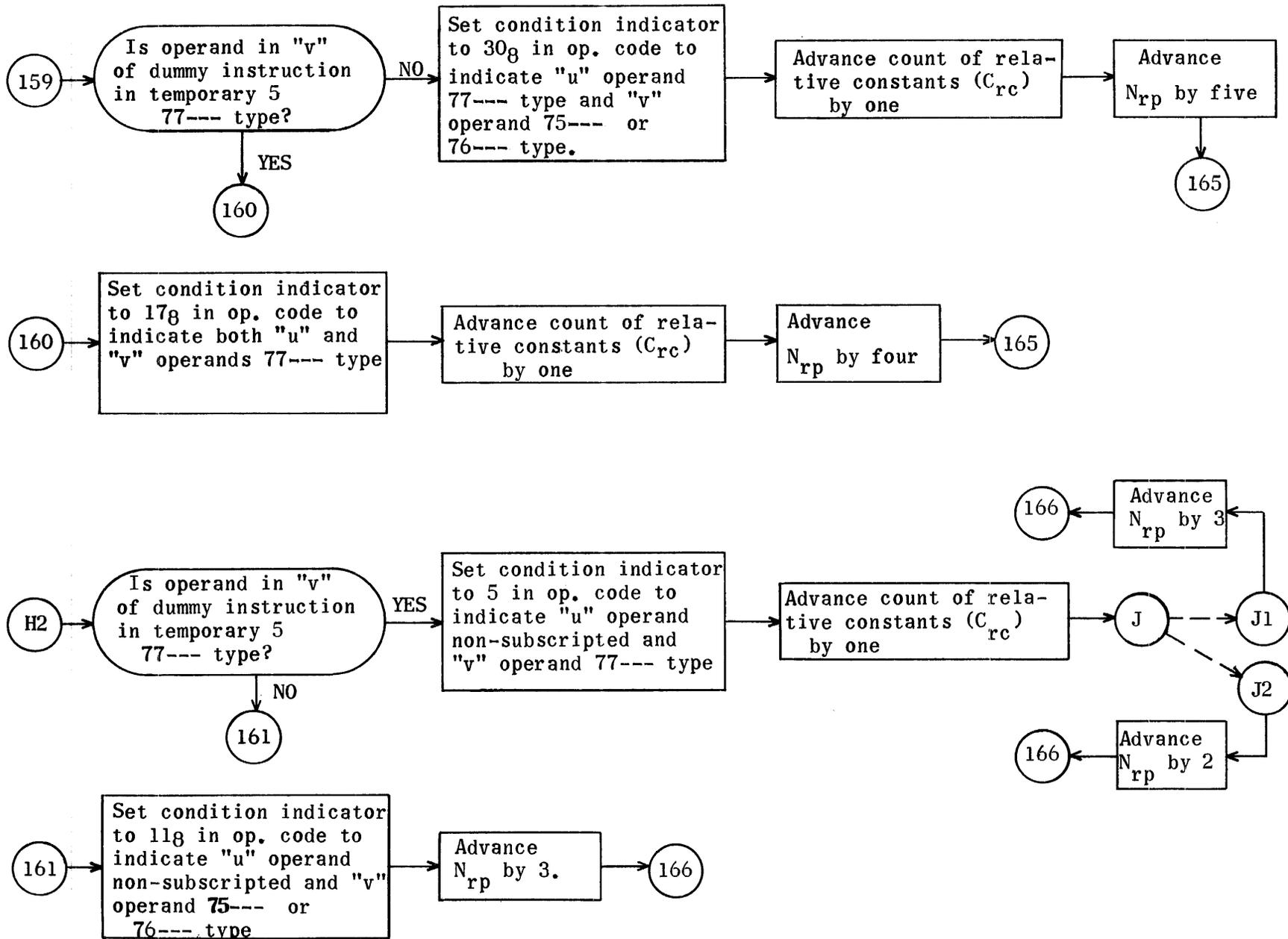


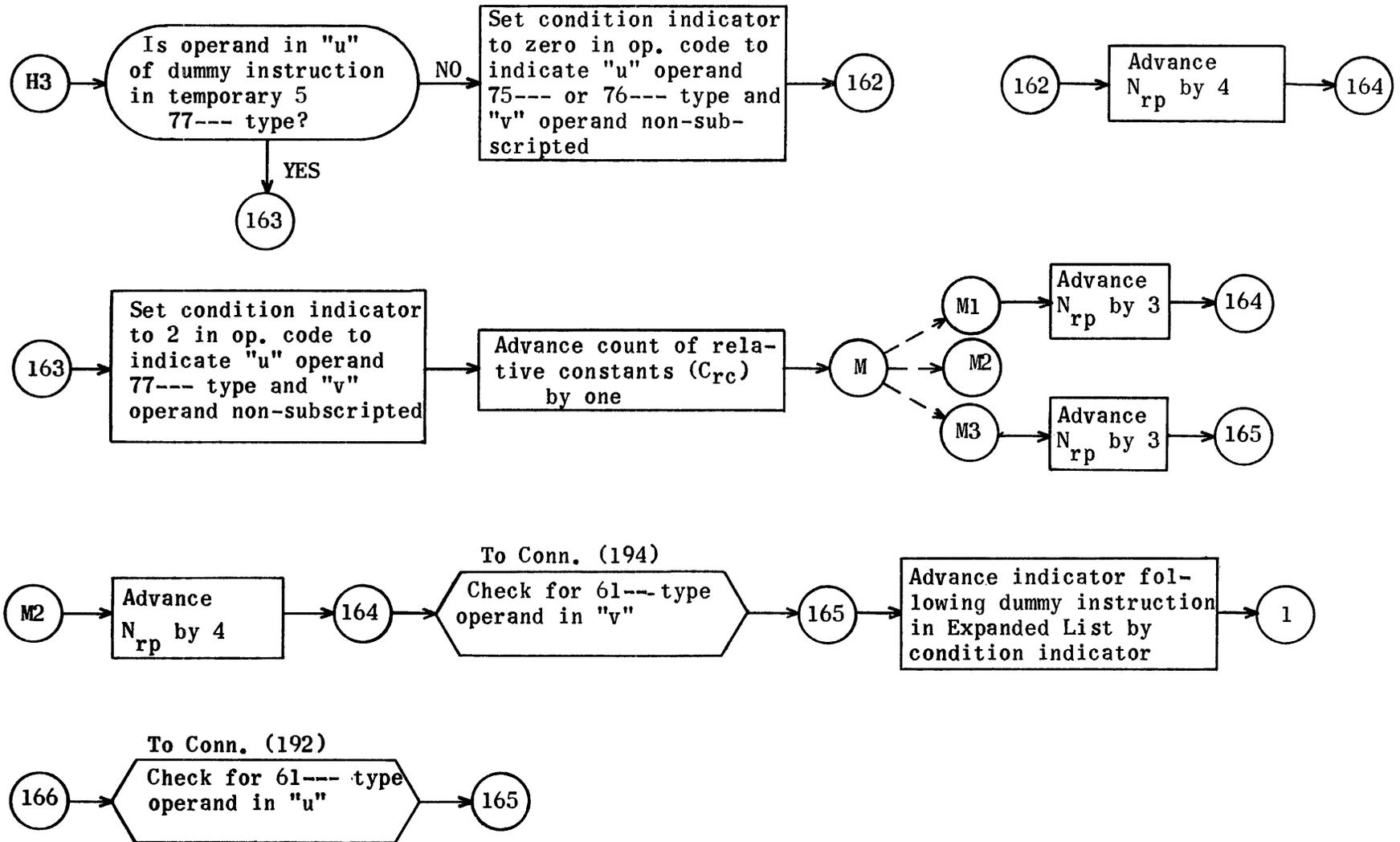
(OS) Subroutine to Sort Operands for Fixed Point Addition and Multiplication



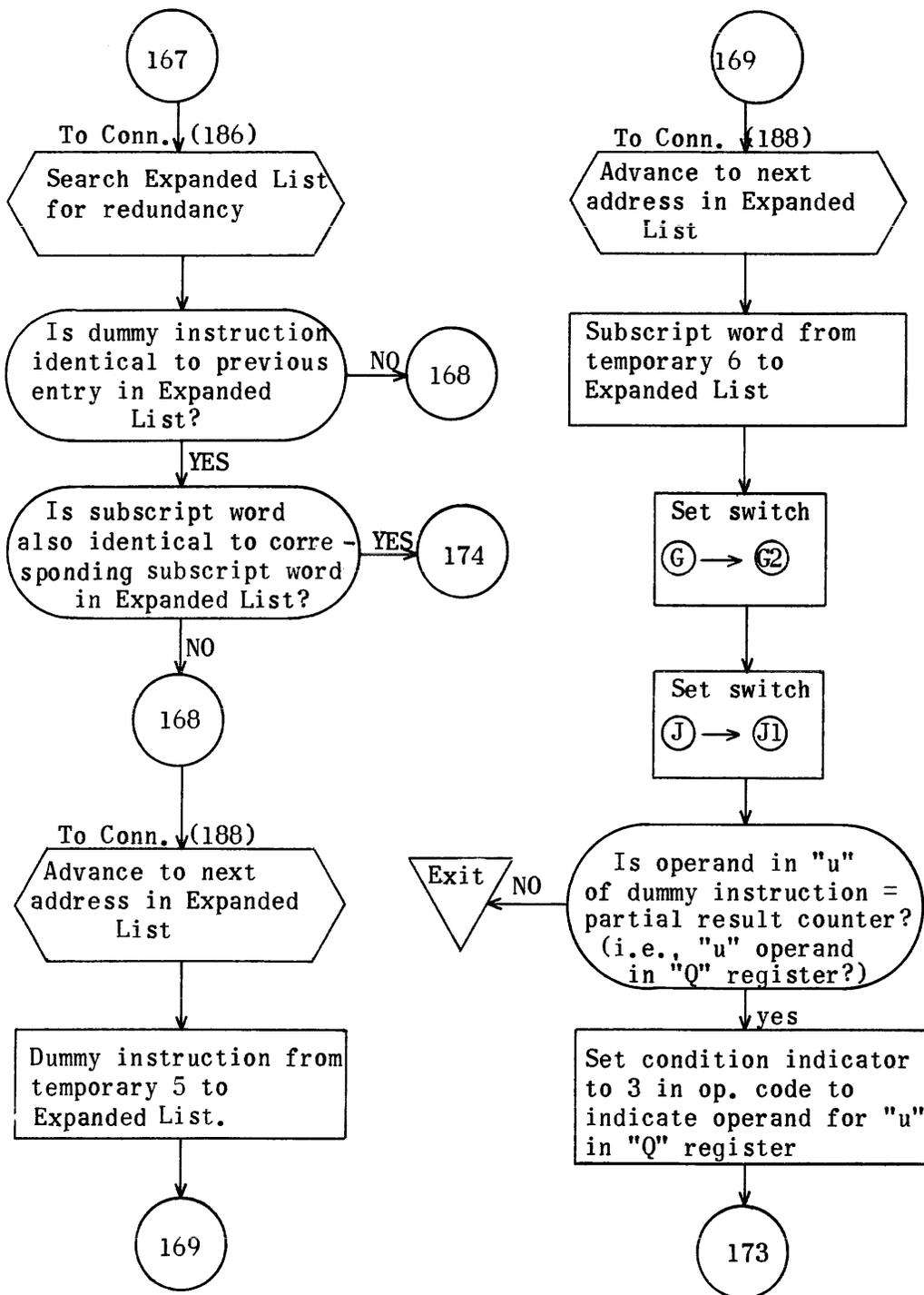
(PN) Set Condition Indicator for Floating Point Operations

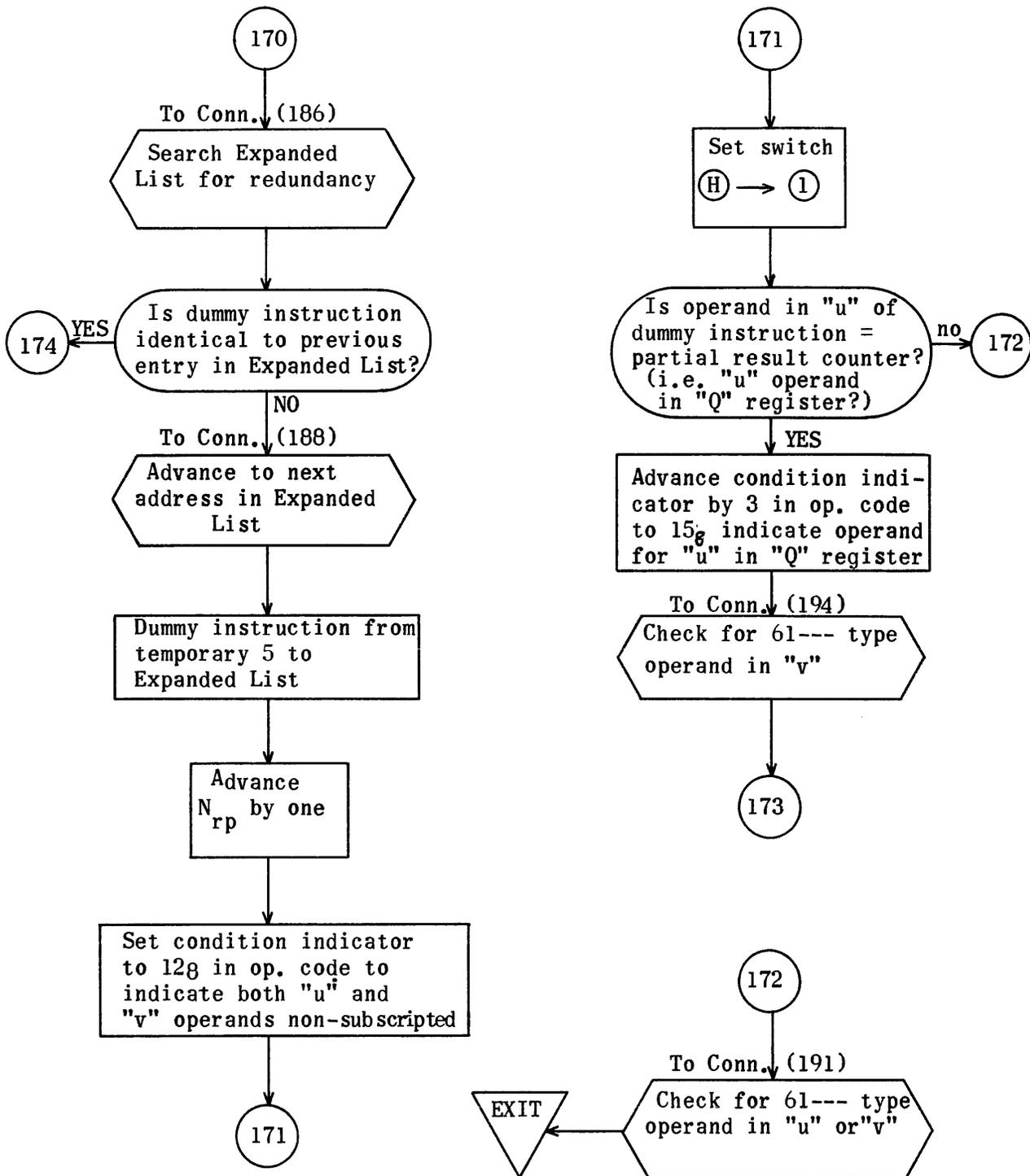


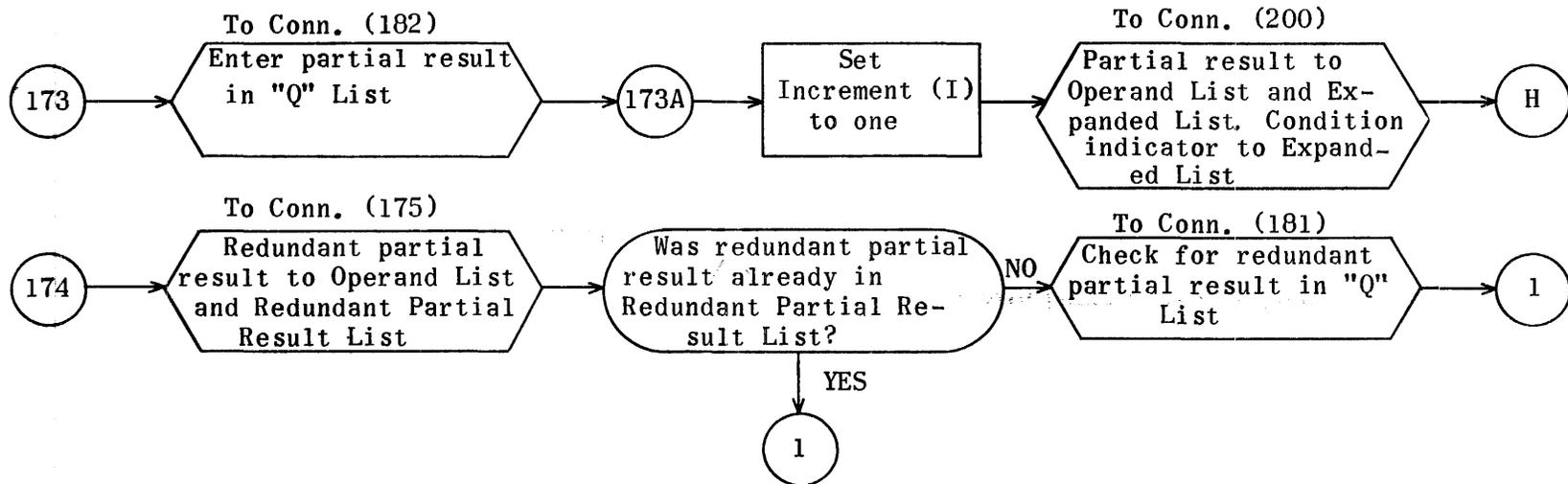




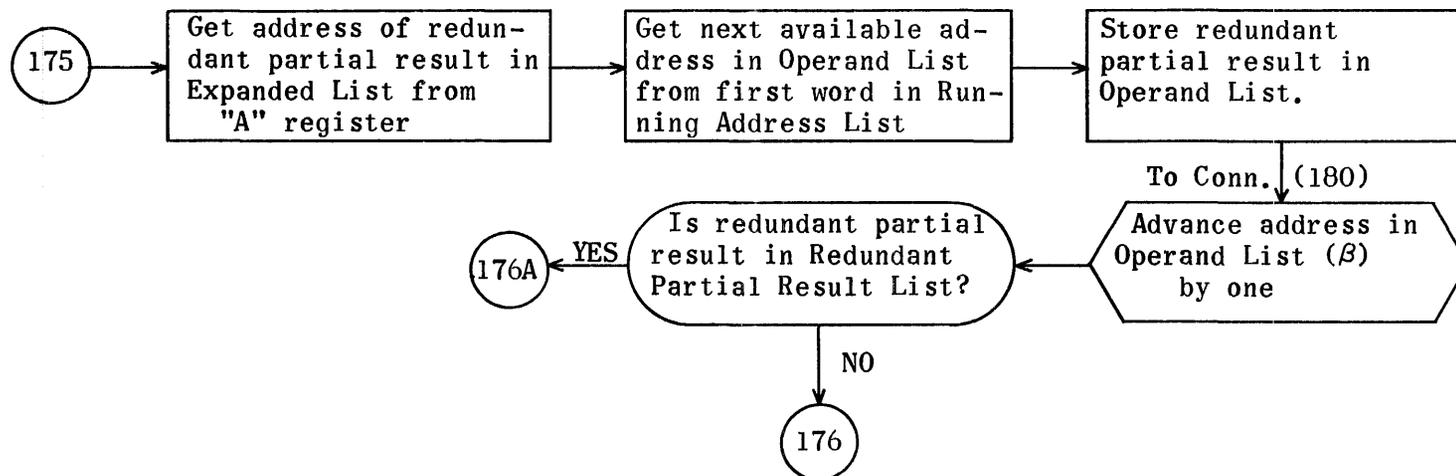
(RR) Subroutine To Check for Redundant Floating Point Binary Operation

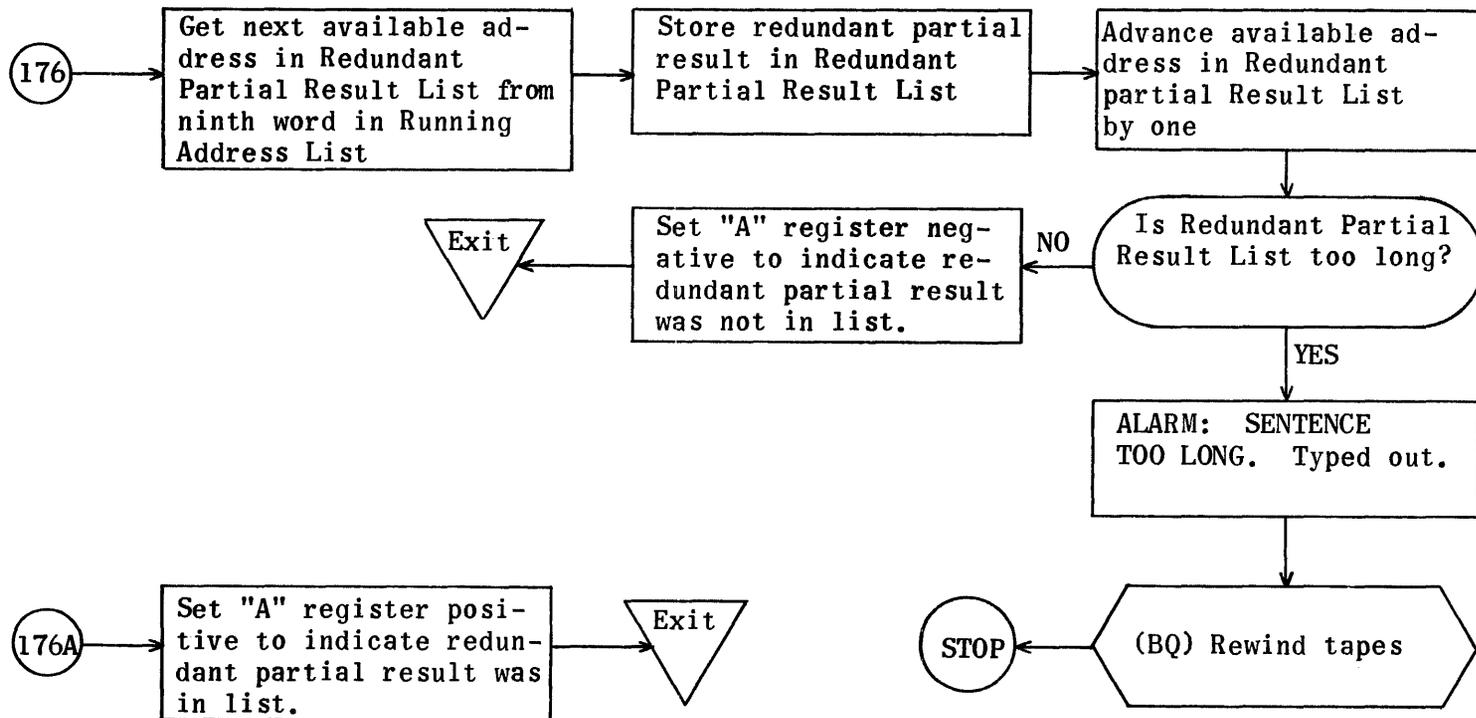




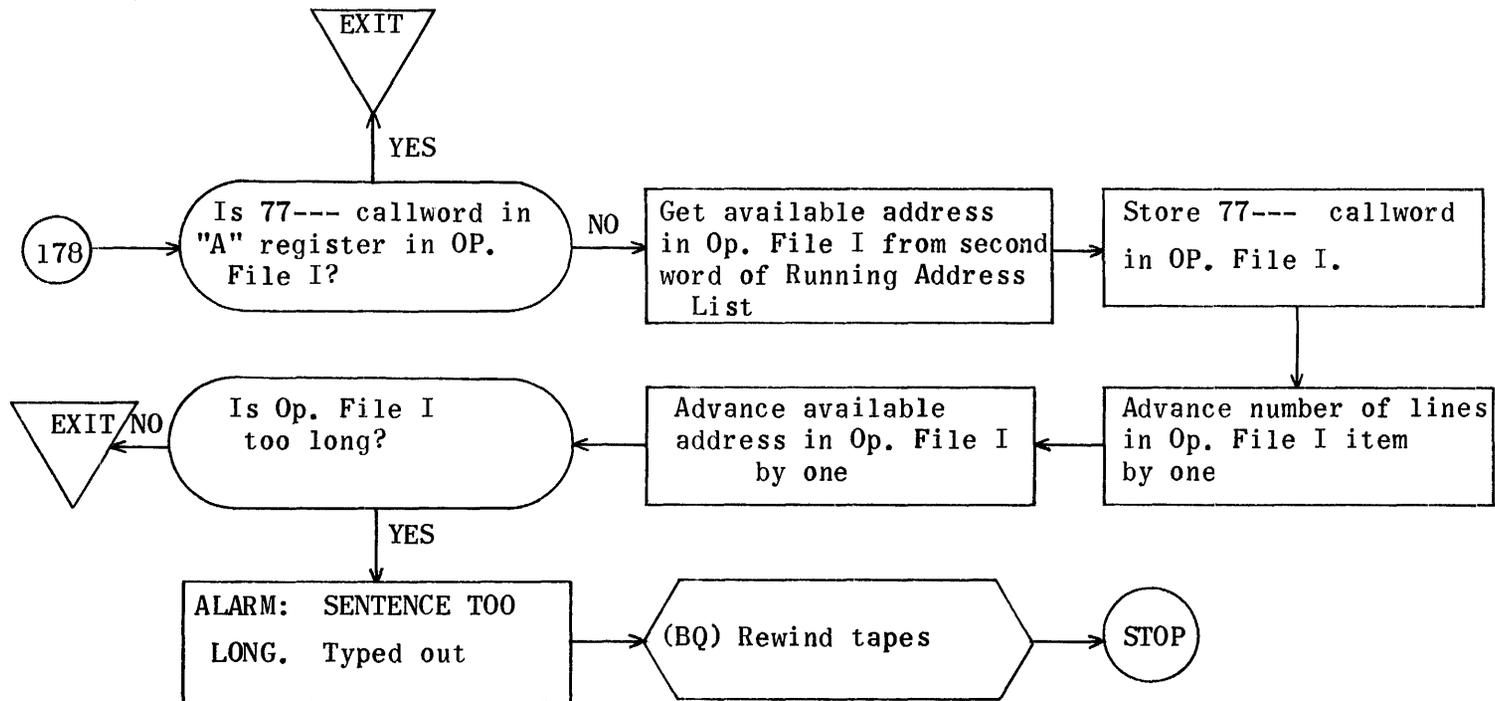


(RS) Subroutine to Store Redundant Partial Result in Operand List and Redundant Partial Result List.

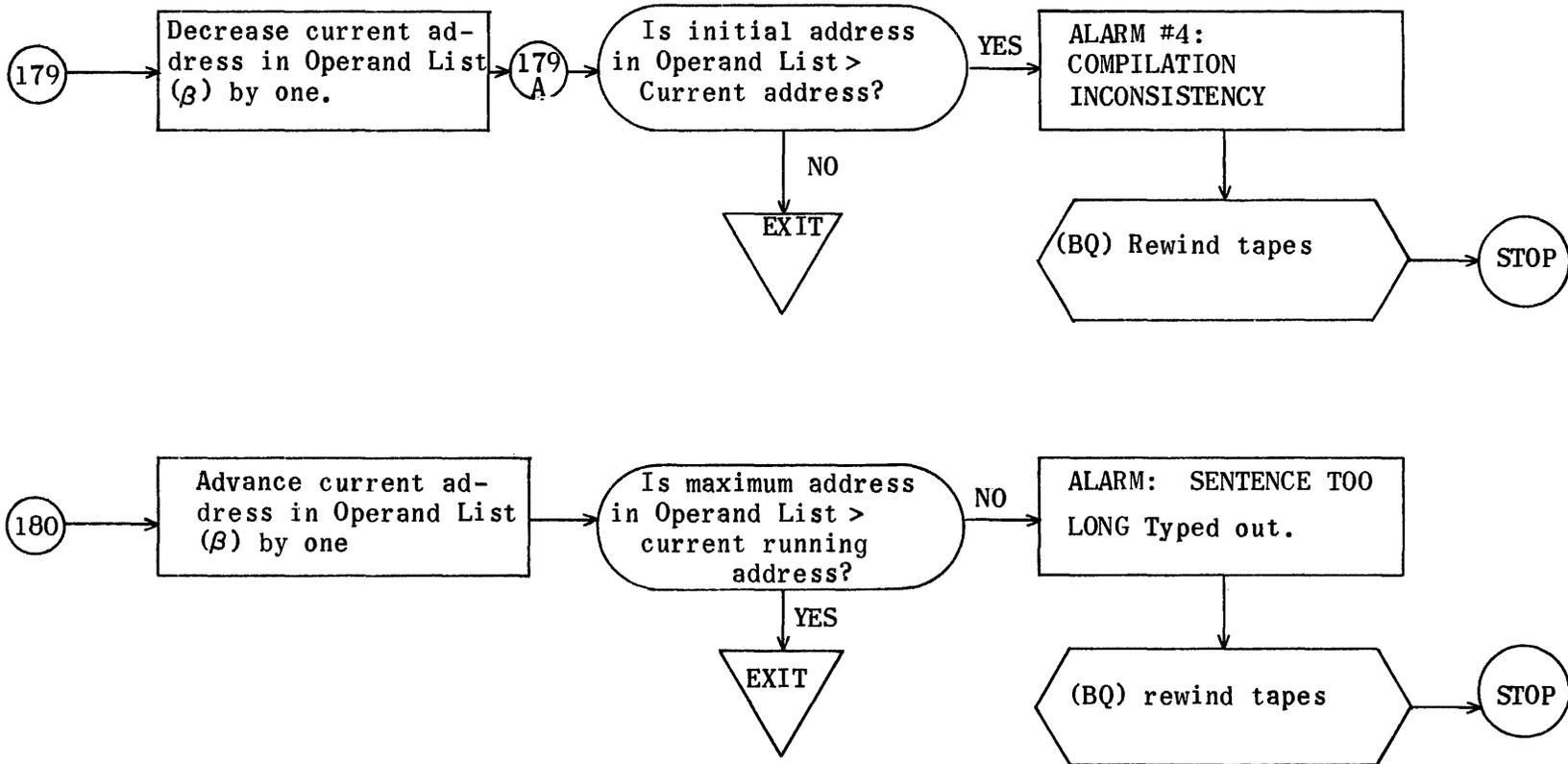




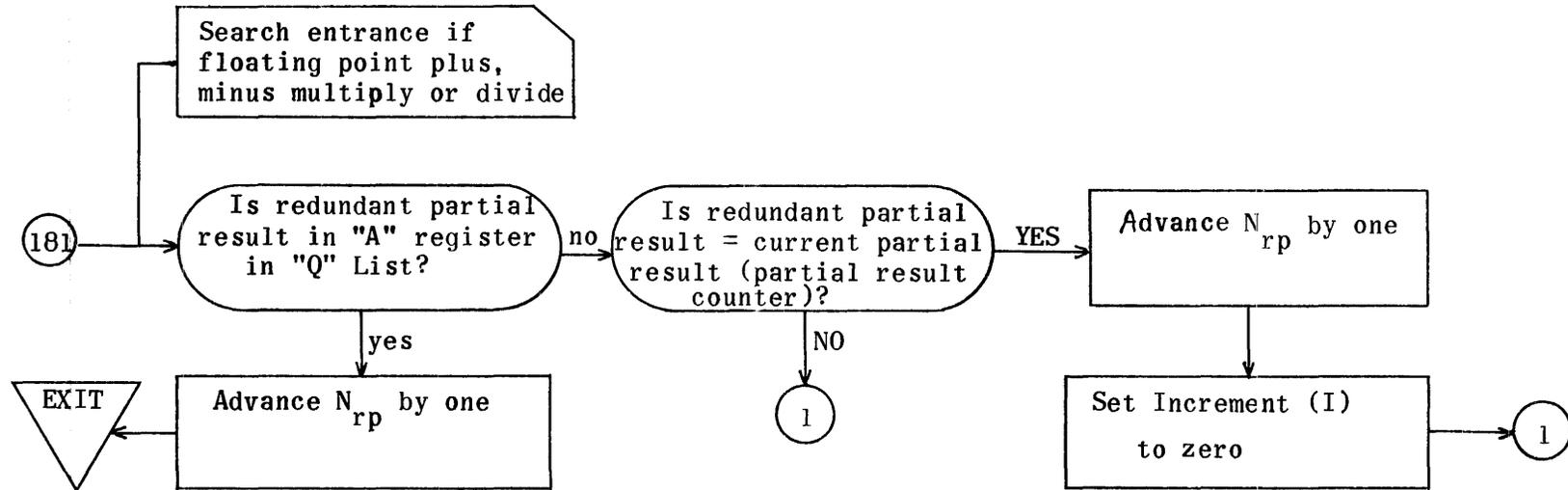
(FS) Subroutine To Store Callword in Op. File 1 Item
 Input-Callword in " " of "A" Register

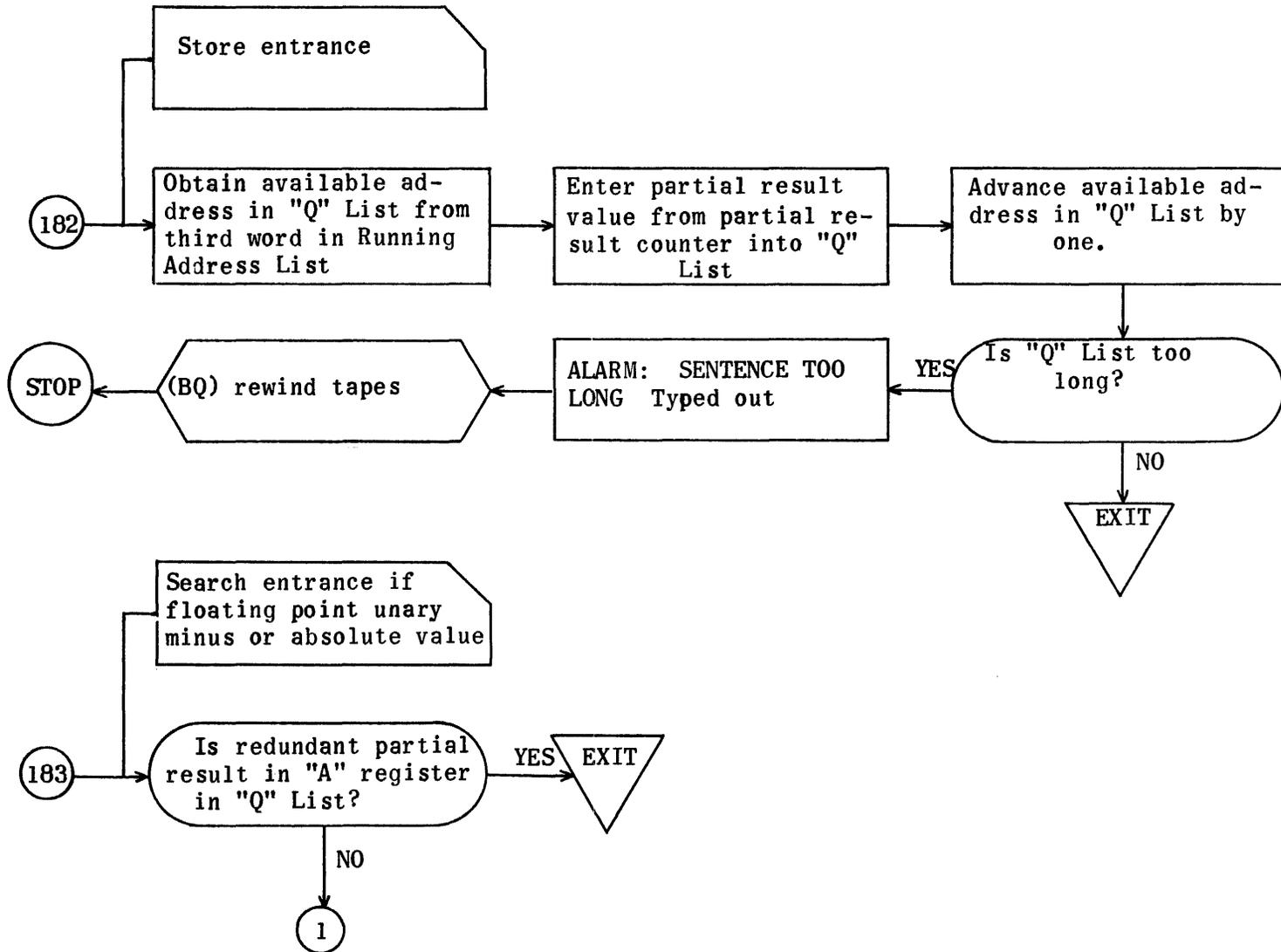


(BR) Subroutine to Advance or Decrease Available Address in Operand List (Beta Routine)



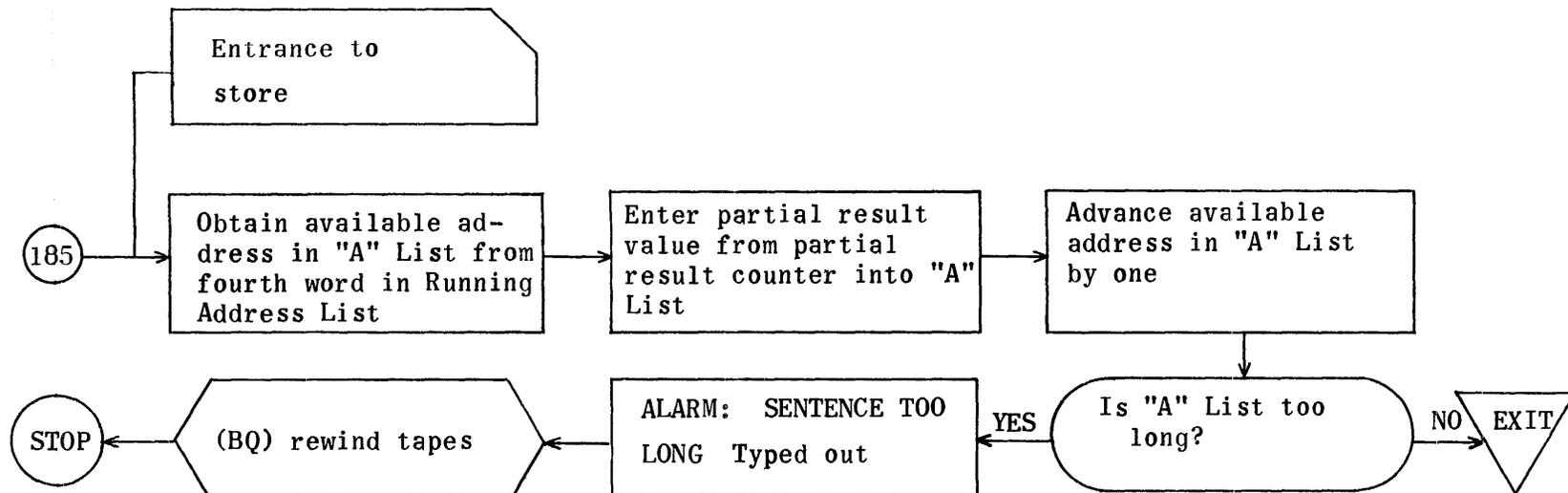
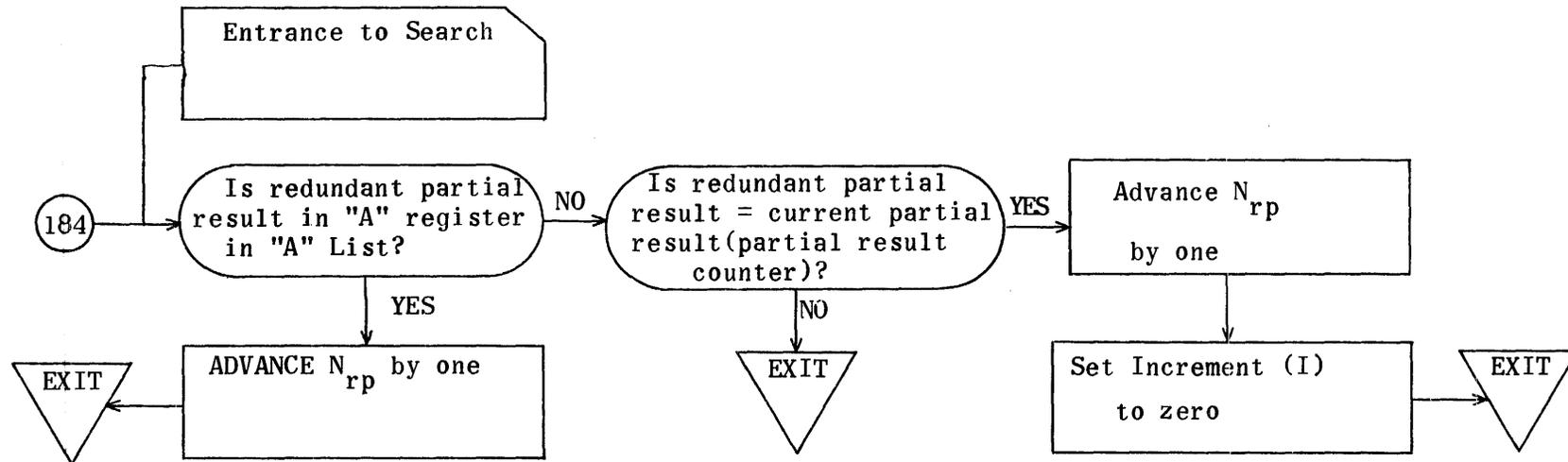
(LQ) Subroutine To Search for or Store Partial Result Symbol in "Q" List
Input-Redundant Partial Result in "A" Register for Search



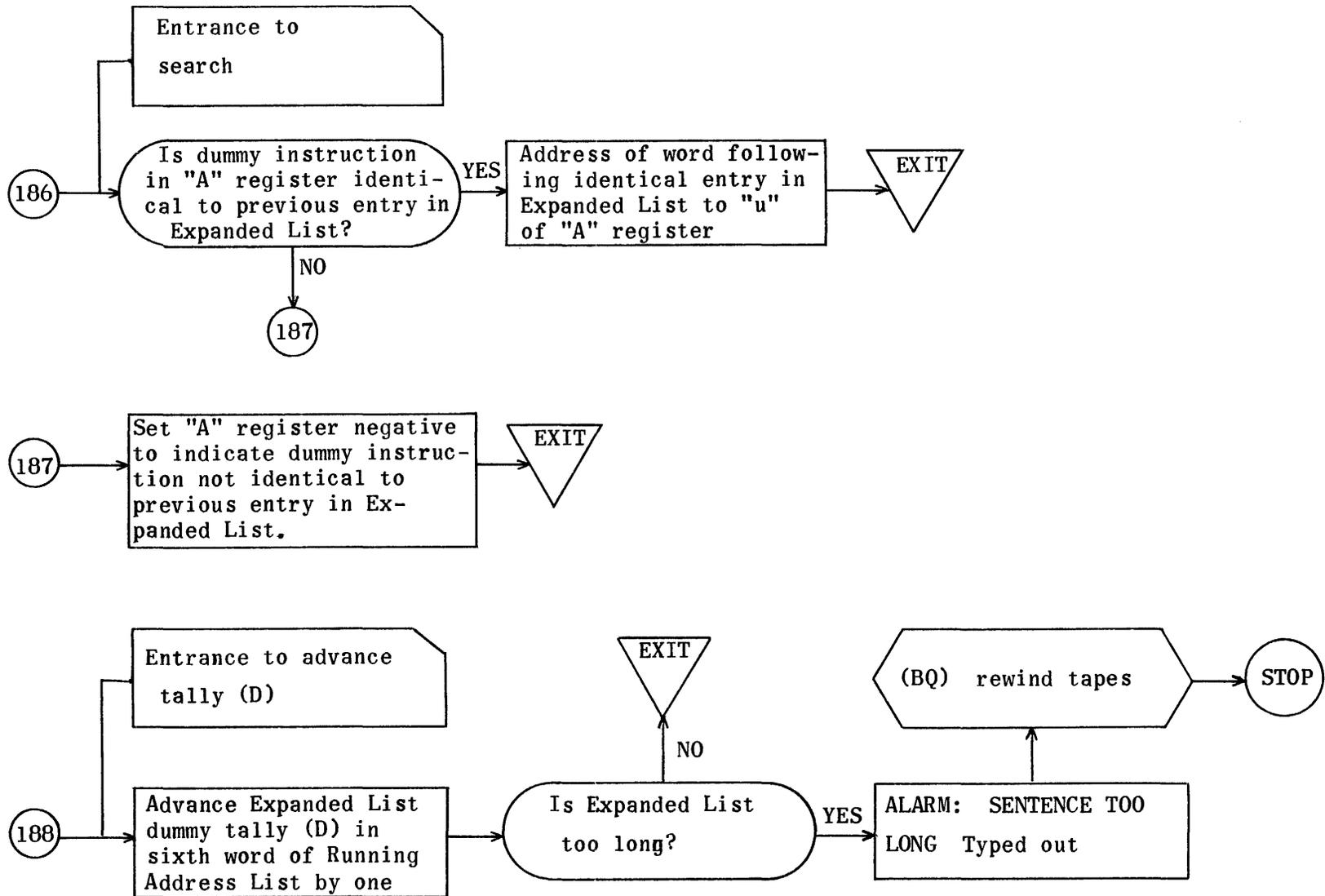


(LA) Subroutine To Search for or Store Partial Result Symbol in "A" List

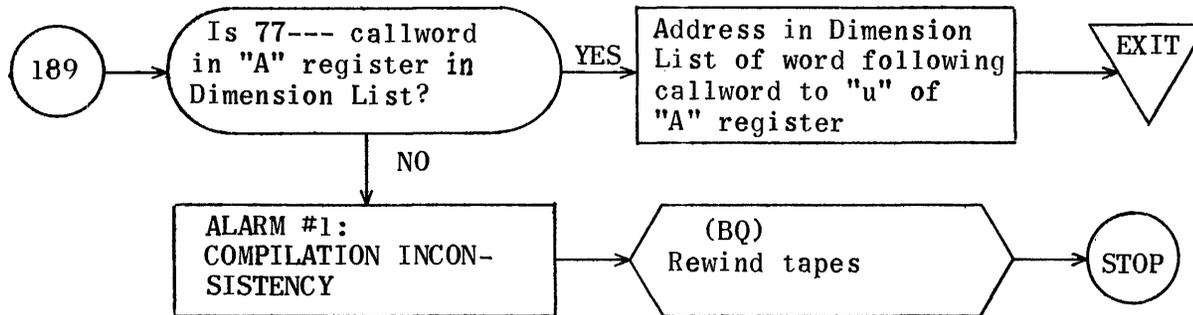
Input—Redundant Partial Result in "A" Register for Search



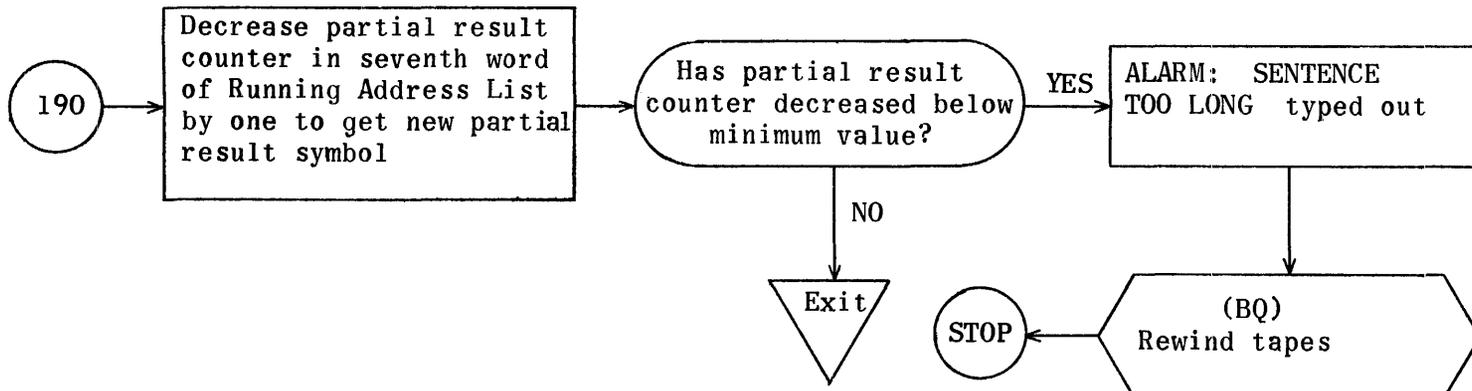
(ES) Subroutine To Search for Dummy Instruction or Advance Dummy Tally in Expanded List
 Input-Dummy Instruction in "A" Register



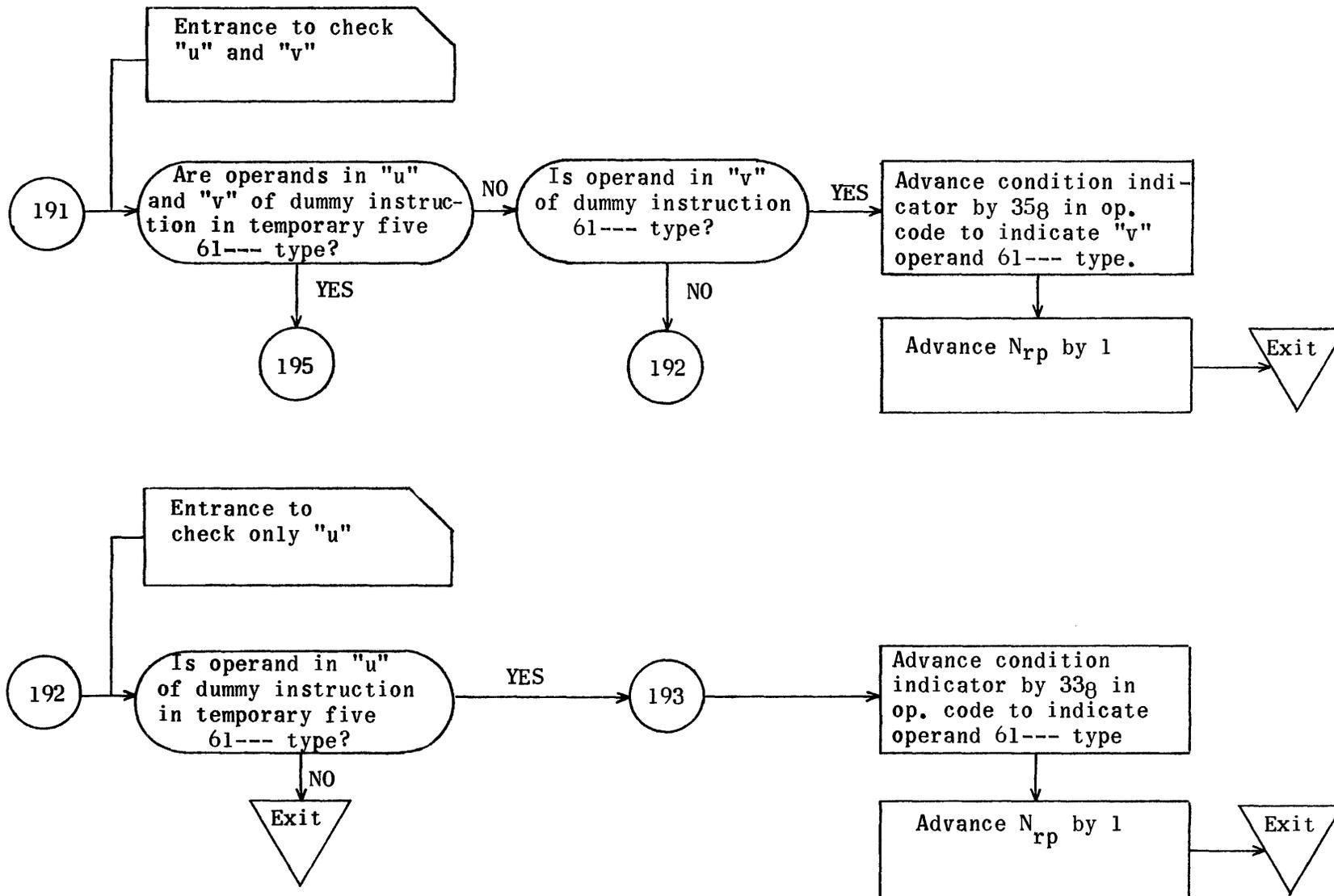
(DS) Subroutine to Search Dimension List
 (Input-Callword in "A" Register)

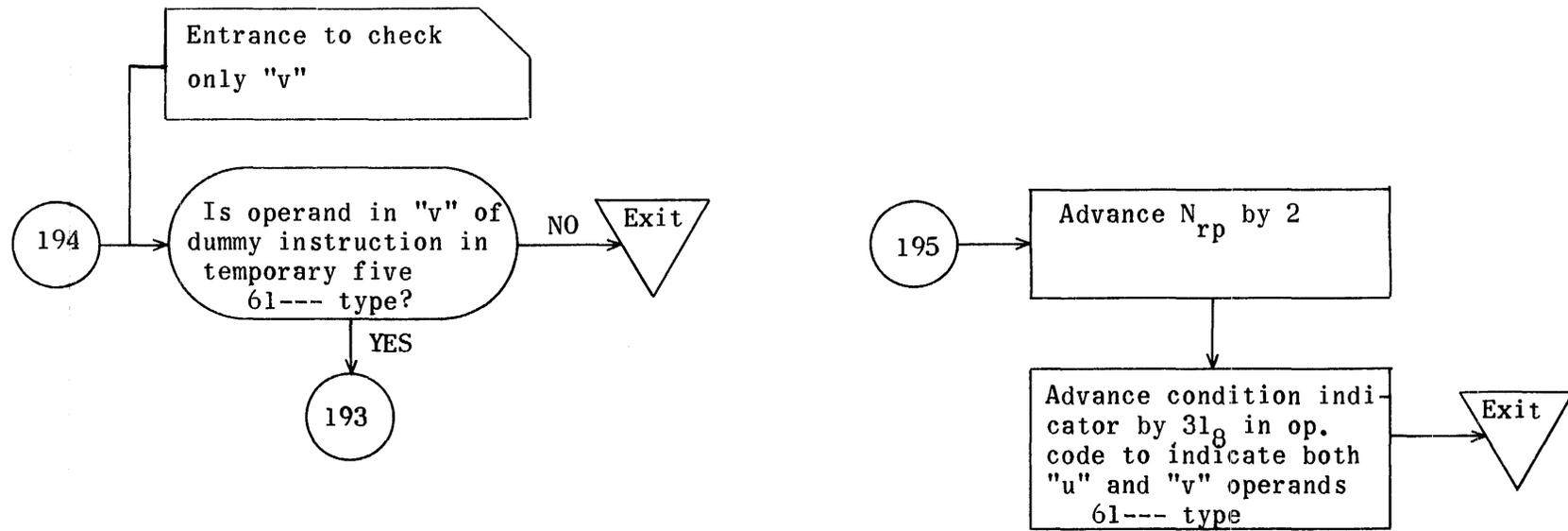


(PR) Subroutine to Decrease and Check Partial Result Counter

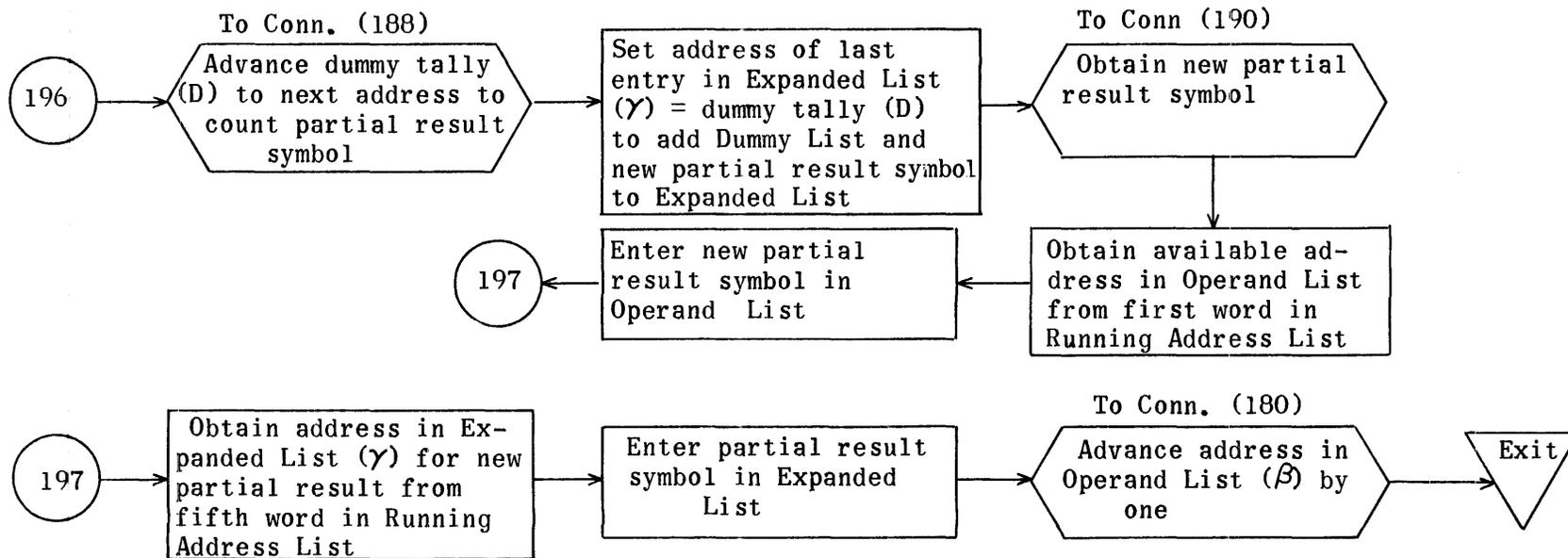


(EK) Subroutine to Check for 61--- Type Operands in Dummy Instruction

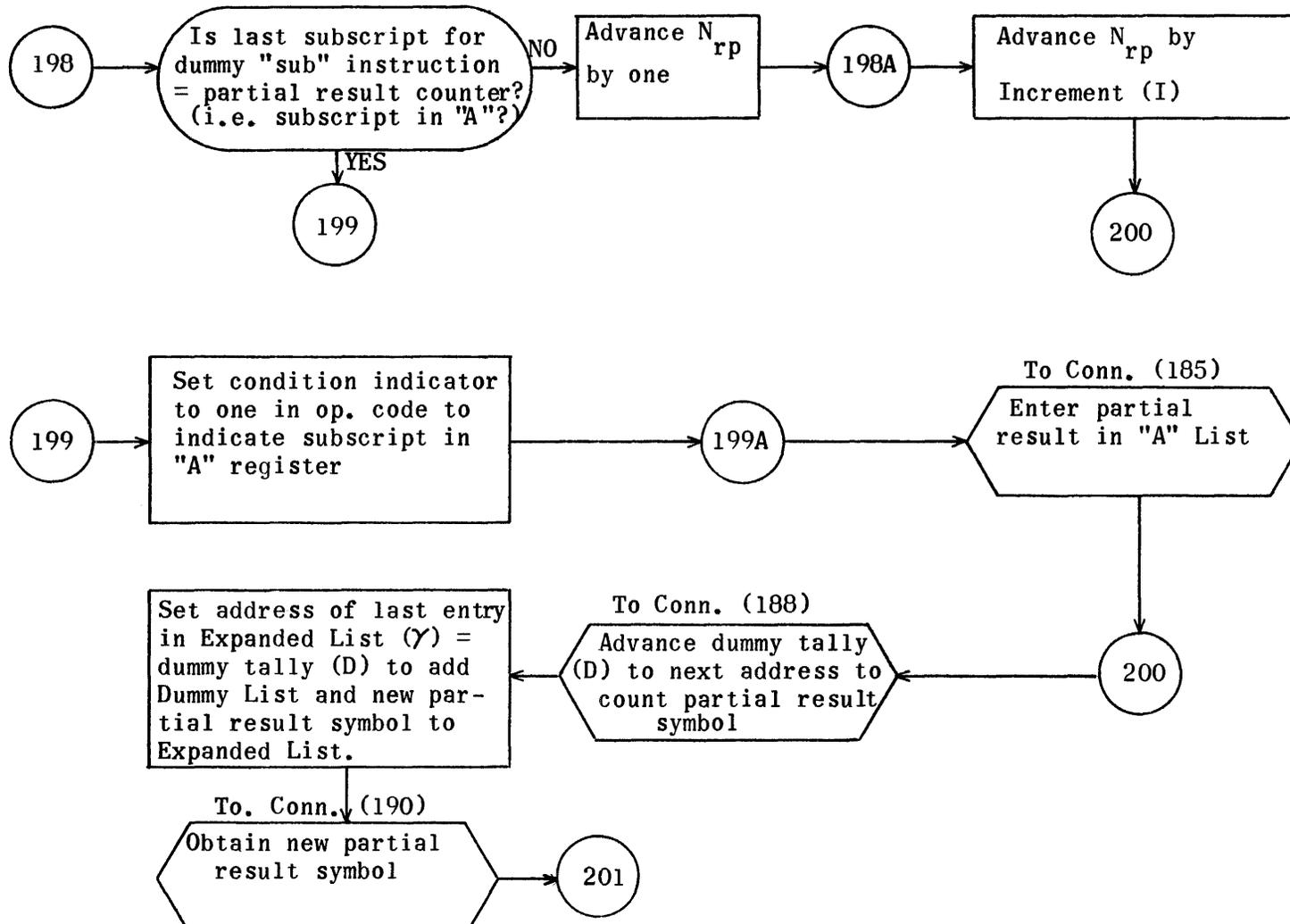


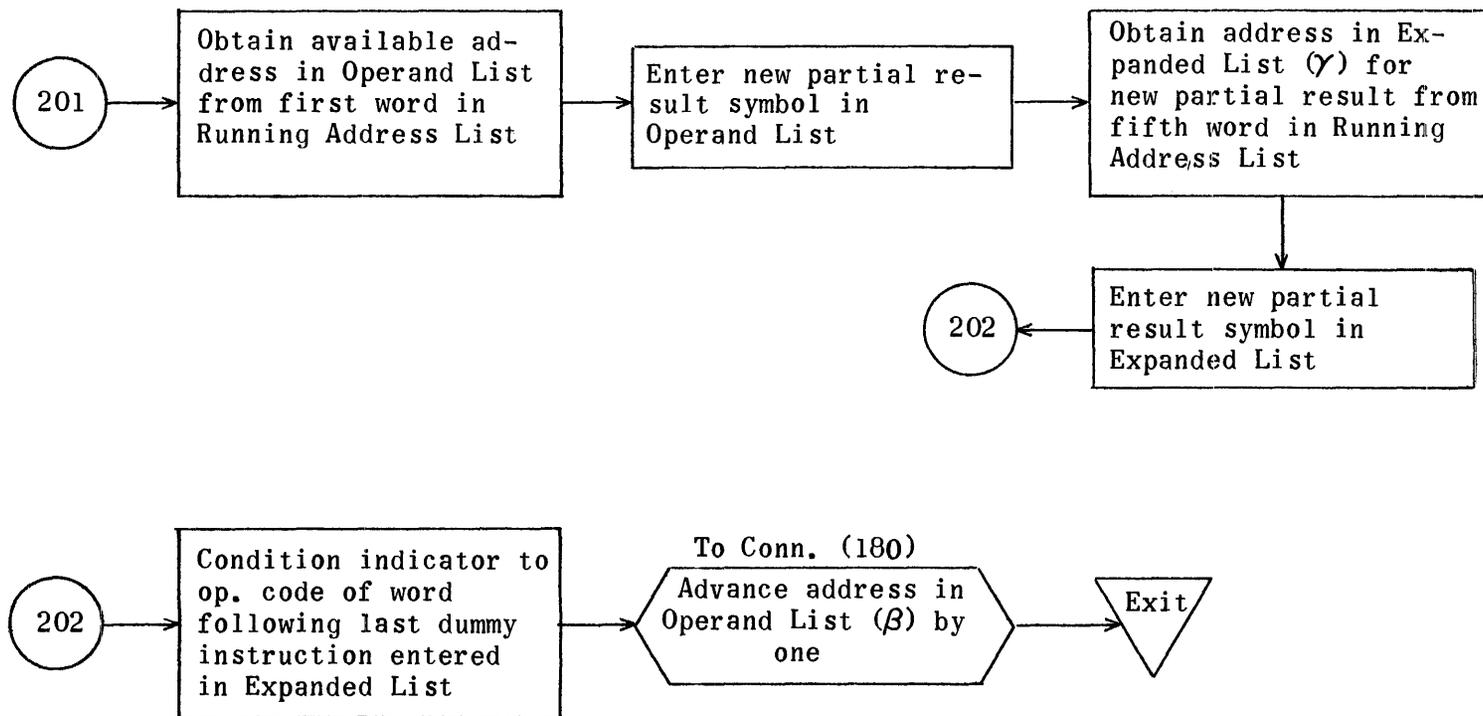


(PP) Subroutine to enter Current Partial Result Symbol in Expanded List and Operand List

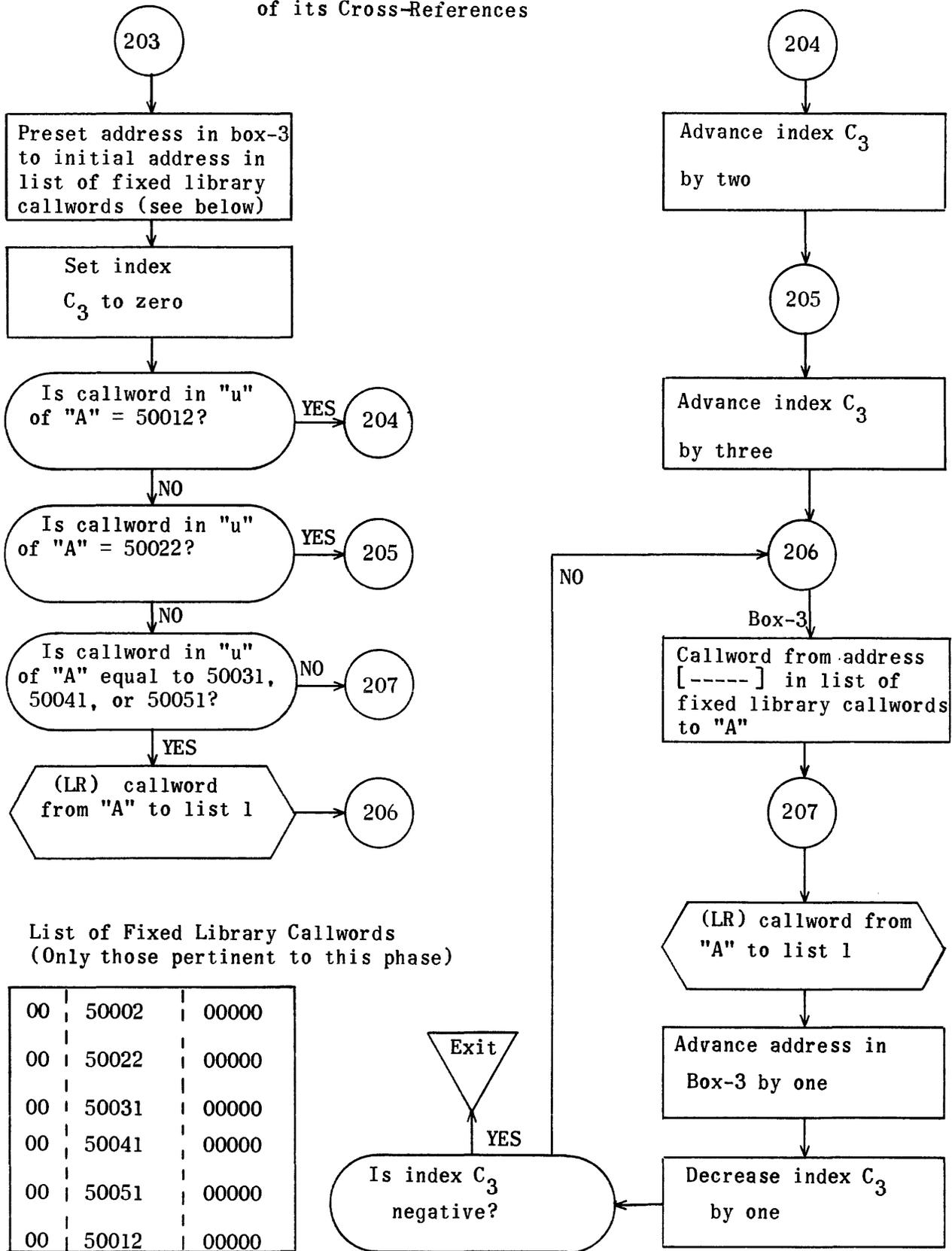


Subroutine to Store Partial Result Symbol for "Sub" Operation in Expanded List and Operand List





(LS) Subroutine to Store in List 1, Callword of Library Routine and, if Fixed Library Routine, Callwords of its Cross-References



List of Fixed Library Callwords
(Only those pertinent to this phase)

00	50002	00000
00	50022	00000
00	50031	00000
00	50041	00000
00	50051	00000
00	50012	00000

Equation No. 2 (Redundancy Check) Coding

Regional Assignments for Equation Redundancy Check Phase

		Region and Address	Name or Description
General Subroutines (not part of this phase) ↓	RE	UP421	Uniprint Routine
	RE	EP537	Alarm Routine
	RE	BQ632	Rewind Tapes Routine
	RE	WA653	Type Alarm Heading
	RE	CW1211	Constant Callword Routine
	RE	LR1465	Build List 1 Routine
Input from Transla- tion	RE	SL2242	Sorted List
Program	RE	BB2512	Setup Redundancy Check Phase (Start)
	RE	SS2544	Check Symbol from Sorted List
	RE	ER2614	End Redundancy Check Phase
	RE	S02633	Subscript Operator (77...type callword)
	RE	SP2715	Subscript Operator (continued)
	RE	SQ2757	Subscript Operator (continued)
	RE	ST3010	Subscript Operator (76...or 75...type call- word)
	RE	LJ3064	Library Operator
	RE	LK3111	Library Operator (continued)
	RE	LL3133	Library Operator (continued)
	RE	LM3161	Library Operator (continued)
	RE	LN3210	Library Operator (continued)
	RE	IP3236	Power Operators (3), (-3), (2), (-2), ($\frac{1}{2}$), ($-\frac{1}{2}$)
	RE	IQ3300	Power Operators (-1), (4to63), (-4 to -63)
	RE	IR3335	Power Operators (continued)
	RE	IS3375	Power Operators (continued)
	RE	FD3454	Floating Point Divide and Subtract Oper- ators
	RE	FP3513	Floating Point Plus and Multiply Opera- tors
	RE	P03544	Fixed Point Plus Operator
	RE	M03570	Fixed Point Multiply Operator
	RE	N03617	Fixed Point Subtract Operator
	RE	D03651	Fixed Point Divide Operator
RE	FN3700	Floating Point Unary Minus and Absolute Value Operators	
RE	NF4014	Fixed Point Unary Minus and Absolute Value Operators	
RE	NE4041	Fixed Point Unary Minus and Abs. Val. (continued)	
RE	EE4051	Storage Operator (space-period)	

Temporaries	RE RE	WT5306 CT5315	Working Temporaries Counters
Lists	RE RE RE RE RE RE RE	RA5550 XQ5561 XA5761 RL6161 EL6261 FL7161 BL7361	Running (current) Addresses in Lists "Q" List "A" List Redundant Partial Result List Expanded List Op. File 1 Item Operand List
Permanent List	RE	DL40102	Dimension List

Equation Redundancy Check Phase

	IA	BB		Setup Redundancy Check Phase
Start	0	MJ	0 [30000]	Exit-Redundancy Phase
	1	TP	IA RA	Preset Running Add. in Operand List
	2	TP	IA1 RA1	Preset Running Add. in Op. File 1 item
	3	TP	IA2 RA2	Preset Running Add. in "Q" List
	4	TP	IA3 RA3	Preset Running Add. in "A" List
	5	TP	IA4 RA4	Preset Running Add. in Expanded List
	6	TP	IA4 RA5	Preset Dummy Tally for Expanded List
	7	TP	IA5 RA6	Preset Partial Result counter
	10	TP	IA6 RA7	Preset Running Tally #lines in Running prog. +1000
	11	TP	IA7 RA10	Preset Running Add. in Red. P.R. List
	12	TU	IA10 SS3	Preset Running Add. in Sorted List→ Initial Add.
	13	TP	SL1 EL1	Line Number to 2nd Word of Expanded List
	14	SP	SL3 17	Callword to "u" of A
	15	TP	A EL	Callword to "u" of 1st word in Exp. List
	16	AT	FC23 FL	Callword to 1st line Op. File 1 item
	17	TP	FC FL1	Zeroize 2nd line Op. File 1 Item
	20	TU	6 DS1	jn from f ₆ to "u" of RP to search Dim. List
	21	TP	FC CT7	Preset increment (I) to Zero
	22	TP	FC CT	Preset Crc to Zero
	23	TP	FC CT1	Preset Crpt to Zero
	24	TP	FC CT2	Preset Crct to Zero
	25	TP	FC WT	Zeroize Temp 0
	26	TP	FC WT1	Zeroize Temp 1
	27	TP	FC CT11	Zeroize index Counter (C ₁)
	30	TP	FC CT12	Zeroize index Counter (C ₂)
	31	MJ	0 SS	
		CA	BB32	

①	0	IA	SS		Check Symbol from Sorted List			
	1	TP	FC	CT10	Zeroize condition Indicator			
	2	TP	FC32	Q	Mask for "v" to "Q"			
②	3	RA	SS3	FC2	Adv. add. in Sorted List → Add. next symbol			
	4	QT	[30000]	A	Symbol from Sorted List → "v" of "A"			
	5	TP	A	WT3	Symbol → "v" of WT3			
	6	TJ	FC112	SS11	50000 > Symbol? (Is this operation Symbol?)			
③	7	TJ	FC43	LJ3	61000 > Symbol? (Is this Library Symbol?)			
	10	TJ	FC61	SQ14	75000 > Symbol? (Is this Non-sub Var. or Const. Sym?)			
	11	MJ	0	S0	Subscripted Variable Symbol (77___, 76___, or 75___)			
	12	LQ	A	17	Symbol → "u" of Q			
	13	QT	FC114	A	Mask rightmost 4 octal digits to "u" of A			
	14	AT	FC113	A	Form <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>MJ</td><td>0[symbol]</td><td>00000</td></tr></table>	MJ	0[symbol]	00000
	MJ	0[symbol]	00000					
	15	RP	30031	SS47	Search List for Operation Symbol			
	16	TJ	SS16	SS16	Jump according to symbol			
	17	MJ	12	FN2	Floating Point Absolute Value			
20	MJ	13	NF2	Fixed Point Absolute Value				
④	21	MJ	20	FP	Floating Point Plus			
	22	MJ	21	PO	Fixed Point Plus			
	23	MJ	30	FD7	Floating Point Subtract			
	24	MJ	31	NO	Fixed Point Subtract			
	25	MJ	32	FN	Floating Point Unary Minus			
	26	MJ	33	NF	Fixed Point Unary Minus			
	27	MJ	50	SS	=			
	30	MJ	52	SS	By			
	31	MJ	60	FP2	Floating Point Multiply			
	32	MJ	61	MO	Fixed Point Multiply			
⑤	33	MJ	70	FD	Floating Point Divide			
	34	MJ	71	DO	Fixed Point Divide			
	35	MJ	100	LJ	General "POWER"			
	36	MJ	101	LJ2	POW > 63 or Non-integral POW < 63 (superscript cases only)			
⑥	37	MJ	120	EE	Storage Operater (space-period)			
	40	MJ	3077	IQ14	Integral Power (4 to 63)			
	41	MJ	3177	IQ16	Integral Power (-4 to -63)			
	42	MJ	4000	IP	Integral Power (3)			
⑦	43	MJ	4100	IP2	Integral Power (-3)			
	44	MJ	5000	IP11	Integral Power (2)			
	45	MJ	5100	IP13	Integral Power (-2)			
	46	MJ	6000	IP22	Power (½)			
	47	MJ	6100	IP24	Power (-½)			
	48	MJ	7100	IQ	Integral Power (-1)			
		CA	SS50					

7A	0	IA ER		End Redundancy Phase
		TP RA7	A	Initial Relative constant Running Address to "A"
	1	TJ LV7	ER6	Number Lines in object program body \leq 1001g?
	2	RJ WA	WA1	No; Type sentence Number
	3	TP TO	UP3	Codeword to Alarm Print
	4	RJ UP2	UP	Alarm; SENTENCE-----TOO LONG.
	5	MJ 0	BQ6	Rewind Tapes and Stop
	6	RS RA2	IA2	#Entries "Q" List to "u" and "v" of "A"
	7	AT FC110	RA	jn for "Q" List Search to Generation Input
	10	RS RA3	IA3	#Entries "A" List to "u" and "v" of "A"
	11	AT FC110	RA1	jn for "A" List Search to Generation Input
	12	RS RA10	IA7	#Redundancy Temps to "u" and "v" of "A"
	13	AT FC110	RA2	jn for Redundant Partial Result Search to Gen. Inp.
	14	TP FC107	RA3	Initial Relative Running Address to Gen. Input
7B	15	TP IA11	RA10	Initial Address in Expanded List +2 to Generation Input
	16	MJ 0	BB	
		CA ER17		

8	0	IA	SO		Subscript Operator (77___ callword)	
	1	TP	A	WT1	7___ symbol to "v" of working Temp #1	
		RJ	ES	ES12	Advance "D" to available dummy inst. address	
	2	TV	A	WT4	Address of dummy inst. to "v" of Temp 4	
9	3	TV	A	S04	Address for dummy inst. to "v" of NI	
	4	TP	RC1	[30000]	Dummy "sub" instruction to Dummy List (D)	
	5	TP	WT3	A	7___ type symbol to "A"	
	6	TJ	FC53	ST	Symbol 77___ ? (i.e. 77000> A)	
	7	RJ	FS	FS1	Yes, store symbol in Op. File 1.	
	10	TP	WT3	A	77___type symbol to "A"	
	11	RJ	DS	DS1	Search Dimension List for symbol (Address of next word in "u" of A)	
	12	TU	A	S013	Address of modulus to "u" of next inst.	
	13	TU	[30000]	WT	Modulus to "u" of Temp 0	
	14	SP	WT	71	Modulus to "v" of A _R	
	15	RJ	CW	CW1	Store modulus in constant pool (callword in "u" of A)	
	16	TV	S04	S017	Address of Dummy inst. in Dummy List (D) to "v" of NI	
	17	TU	A	[30000]	Callword of Modulus to "u" of Dummy instruction	
	20	TU	S013	S021	Address of # S.S. to "u" of NI	
	10	21	TV	[30000]	CT11	# of subscripts to index counter C _i
		22	TP	FC1	A	1 in "v" to "A"
		23	EJ	CT11	SQ	# Subscripts = one?
24		TV	S04	S025	Address of dummy instruction to "v" of NI	
11	25	TV	CT11	[30000]	# Subscripts to "v" of dummy instruction	
	26	SP	CT11	17	#Subscripts to "u" of "A"	
	27	AT	CT11	Q	#Subscripts to "u" and "v" of Q	
	30	RS	RA	Q	Decrease add. in Operand List (β) by #S.S. in "u" and "v"	
	31	RJ	BR	BR2	Has β decreased beyond lower limit.	
	32	TU	A	S040	Address of first s.s. to "u" of TP	
	33	TV	RC31	S042	Preset switch for multiplier in "v"	
	34	TU	S013	S046	Preset address of multiplier	
	35	RS	CT11	FC1	Decrease index counter by 1 in "v"	
	36	RJ	ES	ES12	Advance D to next address in Dummy List	
11	37	TV	A	S040	Address for subscript in Dummy List to "v" of NI	
	40	TP	[30000]	[30000]	Subscript from Operand List to Dummy List in "v"	
	41	RA	S040	FC2	Advance address to next s.s. in Operand List	
	42	IJ	CT11	[30000]	All subscripts transferred to Dummy List?	
	43	MJ	0	SP	Yes	

12

44	TV	RC32	S042	Preset switch for multiplier in "u"
45	RA	S046	FC2	Advance "u" of NI by one (Add. of Mult.)
46	TV	[30000]	WT1	Multiplier to "v" of working Temp.
47	TP	WT1	A	Multiplier to "v" of A
50	RJ	CW	CW1	Store multiplier in constant pool (call- word in "u" of A)
51	TV	S040	S052	Address of subscript in Dummy List to "v" of NI
52	TU	A	[30000]	Multiplier to Dummy List with Corres. Subscript
53	MJ	0	S036	
54	TU	S046	S055	Address of Multiplier in Dim. List to "u" of NI
55	TU	[30000]	WT	Multiplier to "u" of Working Temp.
56	SP	WT	25	Multiplier to "v" of A1
57	LT	0	A	Multiplier to "v" of Ar
60	TV	RC31	S042	Preset switch for multiplier in "v"
61	MJ	0	S050	
	CA	S062		

	IA	SP		Subscript Operator (continued)	
13	0	SP	S04	17	Address of Dummy Inst. to "u" of A
	1	TU	A	SP2	Address of Dummy Inst. to "u" of NI
	2	TP	[30000]	A	Dummy inst to A
	3	TV	A	CT11	#Subscripts to index counter C ₁
	4	TV	A	CT12	#Subscripts to index counter C ₂
	5	RJ	ES	ES1	Search Expanded List for Redundancy
	6	SJ	SP22	SP7	Is dummy inst. redundant? yes to SP7
	7	TP	FC35	Q	Mask for "u" and "v" to "Q"
	10	SS	FC2	0	Add. of prev. entry in Exp. List to "u" of A
	11	TU	A	SP20	Add. of prev. entry in Expanded List to "u" of QT
	12	TU	SP2	SP21	Add. of Dummy inst. to "u" of EJ
	13	IJ	CT11	SP16	All subscripts compared for redundancy?
	14	RA	SP20	FC2	Yes
	15	MJ	0	SQ20	
	16	RA	SP20	FC2	Adv. "u" of QT → Add. of next s.s. in Exp. List
	17	RA	SP21	FC2	Adv. "u" of EJ → Add. of next s.s. in Dummy List
	20	QT	[30000]	A	Subscript from Expanded List → A
	21	EJ	[30000]	SP13	S.S. in Dummy List = S.S. in Expanded List?
14	22	TP	CT12	A	#Subscripts → A
	23	EJ	FC23	SP26	#Subscripts = 2?
	24	EJ	FC57	SP31	#Subscripts = 3?
	25	MJ	0	SP34	Assume four subscripts
15	26	RA	RA7	FC24	Adv. Nrp by 3
	27	TP	FC6	CT10	Set Cond. Ind. → 2(2 subs w/s.s. not in A)
	30	MJ	0	SP36	
16	31	RA	RA7	FC25	Adv. Nrp by 4
	32	TP	FC10	CT10	Set Cond. Ind. → 4(3subs. w/s.s. not in A)
	33	MJ	0	SP36	
17	34	RA	RA7	FC26	Adv. Nrp by 5
	35	TP	FC70	CT10	Set cond. Ind. → 6(4 subs. w/s.s. not in A)
	36	SP	RA5	17	Add. of last entry in Exp. List → "u" of A
	37	TU	A	SR	Address of Last Subscript → "u" of TV
	40	MJ	0	SQ12	
		CA	SP41		

		IA	SQ		Subscript Operator (continued)
18	0	RJ	BR	BR1	Decrease address in Operand List (β) by 1 in "u" and "v"
	1	TU	RA	SQ3	Address of last operand in Oper. List \rightarrow "u" of TV
	2	TV	S04	SQ3	Address of Dummy instruction \rightarrow "v" of NI
	3	TV	[30000]	[30000]	Subscript \rightarrow "v" of dummy instruction
	4	SP	S04	17	Address of Dummy inst. \rightarrow "u" of A
	5	TU	A	SQ6	Address of Dummy inst. \rightarrow "u" of NI
	6	TP	[30000]	A	Dummy instruction \rightarrow A
	7	RJ	ES	ES1	Search Expanded List for redundancy
	10	SJ	SQ26	SQ20	Is dummy inst. = prev. entry in Expanded List?
	20	11	TU	SQ6	SR
12		RJ	SR25	SR	P.R. Value \rightarrow Oper. List and Exp. List; Cond. Ind. \rightarrow Exp. List
21	13	TP	FC3	CT7	Set increment (I) \rightarrow one in "u" and "v"
	14	TV	RA	SQ15	Available address in Operand List (β) \rightarrow "v" of NI
19	15	TP	WT3	[30000]	Operand Symbol \rightarrow Operand List
	16	RJ	BR	BR4	Advance address in Operand List (β) by 1 in "u" and "v"
	17	MJ	0	SS	Return to pick up next symbol in Sorted List
	20	TP	RA4	RA5	Set $D = \gamma$ (delete Dummy List from Expanded List)
	21	RJ	RS	RS1	Redundant P.R. \rightarrow Operand List and Red. P.R. List
	22	SJ	SQ23	SQ24	Was P.R. previously entered in Redundant P.R. List?
	23	TP	WT1	A	No, Redundant Partial Result to A
	24	RJ	LA	LA1	Redundant P.R. in "A" List? (If yes, Advance Nrp by one)
	25	MJ	0	SQ14	
	26	RA	RA7	FC4	Advance Nrp by 2
27	TP	FC	CT10	Set Cond. Ind \rightarrow zero (1 subs. w/s.s. not in A)	
30	MJ	0	SQ11		
	CA	SQ31			

22	0	IA	ST		Subscript Operator (76___ or 75___ CW)
	1	TJ	FC60	ST45	Symbol 76___ ? (i.e. 76000 > A)
	2	TP	FC73	Q	Mask for 3rd octal digit of "v" → Q
		QT	WT3	CT11	#s.s. → 3rd octal digit of "v" of Counter C ₁
	3	LQ	CT11	36	#s.s. → "v" of index counter C ₁
	4	TP	FC74	Q	Mask for rightmost 2 octal digits of "v" → Q
	5	QT	WT3	A	Mask Rel. Location in Ps. Op. Input from 76___ callword
	6	AT	LV11	WT2	Add callword of pseudo op. input region (63000)
	7	SA	CT11	17	Callword of Modulus Location → "u" of A
	10	TV	S04	ST11	Preset address of Dummy Instruction in Dummy List (D)
	11	TU	A	[30000]	Callword of Modulus Location → "u" of Dummy Inst.
	12	TP	FC1	A	1 in "v" → A
23	13	EJ	CT11	SQ	#s.s. = 1? yes to SQ
	14	TV	S04	ST15	Add of Dummy Inst. → "v" of NI
	15	TV	CT11	[30000]	#s.s. → "v" of Dummy Inst.
	16	SP	CT11	17	#s.s. → "u" of A
	17	AT	CT11	Q	#s.s. → "u" and "v" of Q
	20	RS	RA	Q	Decrease address in Operand List (β) by #s.s. in "u" and "v"
	21	RJ	BR	BR2	Has β decreased beyond Lower Limit
	22	TU	A	ST32	Address of first s.s.
	23	RS	CT11	FC1	Reduce "v" of index counter (#s.s.) by one
24	24	SP	WT2	17	Callword of Location of Subs. Variable to "u" of A
	25	TU	A	WT	Callword to working temp.
	26	IJ	CT11	ST30	All subscripts but one transferred to Dummy List
	27	MJ	0	ST40	Yes
	30	RJ	ES	ES12	Advance D → next address in Dummy List
	31	TV	A	ST32	Address for subscript in Dummy List → "v" of NI
	32	TP	[30000]	[30000]	Subscript to Dummy List in "v"
	33	RA	WT	FC2	Adv. "u" of working temp by one → Add. of next mult.
	34	TV	ST32	ST35	
	35	TU	WT	[30000]	Callword of Multiplier Location to Dummy List
	36	RA	ST32	FC2	Adv. "u" address of TP → next s.s. in Operand List
25	37	MJ	0	ST26	
	40	RJ, ES		ES12	Advance D → next address in Dummy List

41	TV	A	ST43	Address for s.s. in Dummy List → "v" of TP
42	TU	ST32	ST43	Address of s.s. in Oper. List → "u" of TP
43	TP	[30000]	[30000]	Last s.s. → Dummy List (no multiplier)
44	MJ	0	SP	
45	TP	FC74	Q	Mask for rightmost 2 octal digits of "v" → Q
46	QT	A	A	Location index in Dummy region from 75100 Cw → Av
47	AT	LV10	WT2	Add callword for function input region (62000)
50	SA	FC1	17	Callword of Modulus Location → "u" of A
51	TV	S04	ST52	Preset Address of dummy "sub" inst. in Dummy List (D)
52	TU	A	[30000]	Callword of Modulus → "u" of dummy inst.
53	MJ	0	SQ	
	CA	ST54		

				Library Operator
27	0	IA LJ TP FC64	WT3	"Gen. Pow." callword (50012) → "v" of symbol Temp.
28	1	MJ 0	LJ3	
	2	TP FC67	WT3	"Var. Exp." CW (50022) → "v" of symbol temp.
29	3	TP FC3	CT3	Set Trp → one (count transfer of control) in "u" and "v"
	4	TP FC	CT4	Set Trc → Zero
	5	TP FC3	CT5	Set Trpt → one (count 10 line for transfer of cont.)
	6	TP FC	CT6	Set Trct → Zero
	7	TP RC4	WT5	Dummy inst → working temp.
	10	TU FC	WT5	Zero → "u" of dummy inst.
	11	TV WT3	WT5	Lib CW → "v" of dummy inst.
30	12	RJ ES	ES12	Adv. "D" to available dummy inst address
	13	TV A	WT4	Add of dummy inst. → "v" working temp.
	14	TV A	LJ15	Add. of dummy inst. → "v" of TP
	15	TP WT5	[30000]	Dummy inst. w/callword → Dummy List
	16	TP FC30	Q	Mask for rightmost octal digit of "v" → Q
	17	QT WT3	CT11	#Arguments → index counter C ₁
	20	TP CT11	CT12	#Arguments → index counter C ₂
	21	TV RC15	LL6	Set switch N to N2
	22	TV RC13	LM24	Set switch S to S1
31	23	IJ CT12	LK	All arguments transferred → Dummy List
	24	MJ 0	LM	
		CA LJ25		

32

	IA	LK		Library Operator (continued)
0	RJ	BR	BR1	Decrease Add. in Oper. List by 1 in "u" and "v"
1	TU	RA	LK2	
2	SP	[30000]	17	Arg. → "u" of A
3	TP	A	WT5	Arg. → "u" of temp. 5
4	TP	FC54	A	74777 → "u" of A
5	TJ	WT5	LL3	Is arg. subscripted? No to LK6
6	SP	RA6	17	P.R. counter → "u" of A
7	TU	WT5	WT	Operand → "u" of working temp.
10	EJ	WT	LL	Operand = P.R. counter? (oper in Q)
11	TP	FC103	Q	No
12	QT	WT5	A	
13	EJ	FC101	LK17	Operand 61---Type?
14	RA	CT3	FC3	No-adv. Trp by one in "u" and "v"
15	RA	CT5	FC3	Adv. Trpt by one in "u" and "v"
16	MJ	0	LL22	
17	RA	WT5	FC115	Adv. indicator by 33g in op. code
20	RA	CT3	FC4	Adv. Trp by two in "u" and "v"
21	MJ	0	LK15	
	CA	LK22		

		IA	LL		Library Operator
33	0	RA	WT5	FC5	Set indicator to 1 in op. code (oper in Q)
	1	TV	RC12	LM24	Set switch (S) to (S2)
34	2	MJ	0	LK14	
	3	RJ	BR	BR1	Dec. add. in Oper. List by 1 in "u" and "v"
	4	TU	RA	LL5	
	5	TV	[30000]	WT5	Subscript → "v" of Arg. word
N1	6	RJ	LL6	[30000]	Switch (N)
	7	TP	FC56	A	76777 → "u" of A
	10	TU	WT5	WT	Oper. → "u" of working temp.
	11	TJ	WT	LL16	Operand 77___ type?
	12	RA	CT3	FC25	Adv. Trp by 4 in "u" and "v"
	13	RA	CT5	FC3	Adv. Trpt by 1 in "u" and "v"
	14	RA	WT5	FC6	Adv. ind. by 2 in op. code (75___or 76___ type arg.)
	15	MJ	0	LL22	
35	16	RA	CT3	FC24	Adv. Trp by 3
	17	RA	CT4	FC3	Adv. Trc by 1
	20	RA	CT6	FC3	Adv. Trct by 1
	21	RA	WT5	FC10	Adv. indicator by 4 in op. code (77___ type arg.)
36	22	RJ	ES	ES12	Adv. "D" to avail. Dummy inst. Add.
	23	TV	RA5	LL24	Add. for Arg. word in Dummy List → "v" of NI
	24	TP	WT5	[30000]	Arg. word → Dummy List
	25	MJ	0	LJ23	
		CA	LL26		

37

	IA	LM		Library Operator
0	SP	WT4	17	Add. of Dummy inst. to "u" of A
1	TU	A	LM2	Add. of Dummy Inst. to "u" of NI
2	TP	[30000]	A	Dummy inst. to A
3	RJ	ES	ES1	Search Exp. List for Redundancy
4	SJ	LM17	LM5	Is dummy inst. redundant? yes to LM5
5	TU	A	LM12	Preset address in Expanded List of first argument
6	TU	LM2	LM15	Preset address of dummy library instruction
7	TP	FC35	Q	Mask for "u" and "v" to Q
10	IJ	CT11	LM12	All arguments compared for redundancy?
11	MJ	0	LM25	
12	QT	[30000]	WT2	Argument from Expanded List to temp. 2
13	RA	LM12	FC2	Advance to next argument in Expanded List
14	RA	LM15	FC2	Advance to next argument in Dummy List
15	QT	[30000]	A	Argument from Dummy List to A
16	EJ	WT2	LM10	Arg. in Dummy List = Arg. in Expanded List

38

17	TP	RA5	RA4	Set _y = D (add Dummy List to Expanded List)
20	RA	RA7	CT3	Adv. Nrp by Trp
21	RA	CT	CT4	Adv. Crc by Trc
22	RA	CT1	CT5	Adv. Crpt by Trpt
23	RA	CT2	CT6	Adv. Crct by Trct
24	MJ	0	[30000]	Switch (S)
25	TP	LM12	A	Address of redundant partial result to "u" of A
26	MJ	0	LN21	
	CA	LM27		

Check for 1st subscript in A	S1	0	IA LN	
		1	RA RA7 CT7	
	S2	2	MJ 0 LN5	
		3	RJ LQ LQ7	
		4	MJ 0 LN5	
	S3	5	RJ LA LA6	
	39	6	TP FC3 CT7	
		7	RJ PP10 PP	
	40	7	TP WT3 A	
		10	RJ FS FS1	
		11	MJ 0 LN23	
	N2	12	TV WT5 WT1	
		13	TP RA6 A	
		14	EJ WT1 LN16	
		15	MJ 0 LL7	
		16	TV RC14 LM24	
		17	RA WT5 FC5	
		20	MJ 0 LL7	
		21	TP RA4 RA5	
		22	MJ 0 RR50	
		23	SP WT3 17	
		24	RJ LS LS1	
		25	MJ 0 SS	
			CA LN26	

Library Operator

Adv. Nrp by Increment (I)

Enter P.R. in "Q" List

Enter P.R. in "A" List

Set increment (I) → one in "u" and "v"

Enter current P.R. in Oper. List and Exp. List

Lib. callword → A

Enter Lib. callword in Op. File 1

Subscript → "v" of working temp.

P.R. counter → A

P.R. counter = subscript?

Set S to S3

Set indicator → 1 in op. code (s.s. in A)

Set D = γ (inst. Red. do not add Dummy List to Exp. List)

Library Routine Callword to List 1

				Power Operators
		IA	IP	(3), (-3), (2), (-2), (½), (-½)
41	0	TP	RC11	WT5
	1	MJ	0	IP3
42	2	TP	RC12	WT5
43	3	RJ	IR15	IR
	4	QJ	IP5	IP7
	5	RA	RA7	FC26
	6	MJ	0	IR16
	7	RA	RA7	FC4
	10	MJ	0	IS33
44	11	TP	RC7	WT5
	12	MJ	0	IP14
45	13	TP	RC10	WT5
46	14	RJ	IR15	IR
	15	QJ	IP16	IP20
	16	RA	RA7	FC25
	17	MJ	0	IR16
	20	RA	RA7	FC3
	21	MJ	0	IS33
47	22	TP	RC13	WT5
	23	MJ	0	IP25
48	24	TP	RC14	WT5
49	25	RJ	IR15	IR
	26	QJ	IP27	IP32
	27	RA	RA7	FC25
50	30	RJ	IP41	IP35
	31	MJ	0	IR16
	32	RA	RA7	FC4
51	33	RJ	IP41	IP35
	34	MJ	0	IS21
	35	TP	FC66	A
	36	RJ	FS	FS1
	37	TP	LS25	A
	40	RJ	LS	LS1
	41	MJ	0	[30000]
		CA	IP42	

		IA	IQ		Power Operators (-1),(4to63),(-4 to -63)
(52)	0	TP	RC17	WT5	Entrance-POW (-1) inst. to temp. 5
	1	RJ	IR15	IR	Check for redundancy
	2	QJ	IQ3	IS21	Is operand subscripted?
	3	RA	RA7	FC4	Advance Nrp by 3 in "u" and "v"
	4	TU	WT5	WT	Operand to working temp.
	5	TP	FC56	A	76777→ "u" of A
	6	TJ	WT	IR27	Is operand 77___type?
	7	TP	FC12	CT10	10→ op. code of cond. ind.
(53)	10	TV	WT5	WT1	
	11	TP	RA6	A	P.R. ocunter → A
	12	EJ	WT1	IS	Operand = P.R. counter? (subscript in "A"?)
	13	MJ	0	IR33	No
(54)	14	TP	RC15	WT5	Entrance-POW (4 to 63)
	15	MJ	0	IQ17	
(55)	16	TP	RC16	WT5	Entrance-POW (-4 to -63)
(56)	17	TV	RC17	IR10	Set switch (T) to (T2)
	20	RJ	IR15	IR1	Check for redundancy
	21	RJ	ES	ES12	Advance dummy tally D by one
	22	TV	RA5	IQ23	Available address in Exp. List → "v" of TP
	23	TP	WT3	[30000]	13___symbol in "v" → Exp. List
	24	QJ	IQ25	IQ27	Is operand subscripted?
	25	RA	RA7	FC27	Advance Nrp by 6 in "u" and "v"
	26	MJ	0	IR16	
	27	RA	RA7	FC24	Advance Nrp by 3 in "u" and "v"
	30	MJ	0	IS33	
(T2)	31	TU	A	IQ32	Address of 13___symbol (POW word) in Exp. List → "u" of TP
	32	TP	[30000]	A	13___symbol (POW word) from Exp. List → A
	33	EJ	WT3	RR50	Is 13___symbol (POWword) also redundant?
	34	MJ	0	IR11	
		CA	IQ35		

				Power Operators (continued)
Redundancy check	(57)	0	IA IR	Set switch (T) to (T1)
	(58)	1	TV RC16 IR10	Check variables and set switch (H)
		2	RJ VC50 VC5	s.s. "yes or no" indicator → working temp.
		3	TP Q WT2	Is there a subscript?
		4	QJ IR4 IR36	Address of s.s. → "v" of NI
		5	TU RA IR5	s.s. → "v" of dummy inst.
		6	TV [30000] WT5	Dummy instruction to "A"
		7	TP WT5 A	Search Exp. List for instruction
		10	RJ ES ES1	Is instruction redundant? (switch T)
	(59)	11	SJ IR11 [30000]	Advance dummy tally by one
		12	RJ ES ES12	Available address in Expanded List → "v"
			TV RA5 IR13	of TP
Operand subscripted		13	TP WT5 [30000]	Instruction to Expanded List
		14	TP WT2 Q	s.s. "yes or no" indicator → Q
	(60)	15	MJ O [30000]	Exit
		16	TU WT5 WT	"u" of dummy inst. → working temp.
		17	TP FC56 A	76777 → "u" of A
		20	TJ WT IR27	Is operand 77___type
		21	TP FC12 CT10	Set cond. ind. → 10 (operand 74___type)
		22	TV WT5 WT1	Subscript to working temp.
		23	TP RA6 A	P.R. counter → A
		24	EJ WT1 IS	P.R. counter = subscript? (is subscript in A)
		25	RA RA7 FC3	Advance Nrp by one in "u" and "v"
	(61)	26	MJ O IR34	
		27	TP FC10 CT10	Set cond. ind. → 4 (operand 77___type)
		30	RA CT FC3	Advance Crc (count of rel. const.) by one in "u" and "v"
		31	TV WT5 WT1	Subscript to "v" of working temp.
		32	TP RA6 A	P.R. counter → A
		33	EJ WT1 IS	Subscript = P.R. counter (is subscript in A)
		34	RA RA7 CT7	Advance Nrp by increment (I)
		35	MJ O IS2	
		36	TV FC WT5	Zero → "v" of dummy inst.
	37	MJ O IR6		
		CA IR40		

					Power Operators (continued)	
62	0	IA	IS		Adv. op. code of cond. ind. by one (s.s. in A)	
		RA	CT10	FC5		
	63	1	RJ	LA	LA6	Enter P.R. value in "A" List
		2	TP	FC3	CT7	Set increment (I) → one
		3	RJ	PP10	PP	New P.R. value → Exp. List and Oper. List
		4	TV	RA4	IS6	Address of P.R. value in Exp. List → "v" of QT
		5	TP	FC36	Q	
		6	QS	CT10	[30000]	Indicator → op. code of P.R. word
		7	TP	FC73	Q	Mask for 3rd octal digit of "v" → Q
		10	QT	WT3	A	3rd octal digit → A
	11	ZJ	IS13	IS12	3rd octal digit = 1 (is this neg. power)	
64	12	MJ	0	IS44		
	13	RA	RA7	FC3	Advance Nrp by one in "u" and "v"	
	14	TP	FC65	A	Floating point one → A	
	15	RJ	CW	CW1	Store floating pt. "one" in constant pool	
	16	TV	IS6	IS17	Address of P.R. word → "v" of NI	
	17	TU	A	[30000]	Callword of fixed const. → "u" of P.R. word	
		20	MJ	0	IS44	
65	21	TU	WT5	WT	Operand → working temp.	
	22	SP	RA6	17	P.R. counter → "u" of A	
	23	EJ	WT	IS30	Operand = P.R. counter? (operand in "Q")	
	66	24	TP	FC	CT10	No, set op. code of cond. ind. → Zero
		25	RJ	EK25	EK6	To 6l___routine ("u" ent.)
		26	RA	RA7	CT7	Advance Nrp by increment (I)
	67	27	MJ	0	IS2	
		30	TP	FC5	CT10	Set op. code of cond. ind. → one
		31	RJ	LQ	LQ7	Enter P.R. value in Q list
	68	32	MJ	0	IS2	
33		RA	RA7	CT7	Advance Nrp by increment (I)	
34		TP	FC	CT10	Set op. code of cond. ind. → zero	
35		TP	FC103	Q	Mask for first two octal digits of "u" → Q	
69	36	QT	WT5	A	First two octal digits of "u" of D → A	
	37	EJ	FC101	IS41	Operand = 6l___type?	
	40	MJ	0	IS2		
	41	RA	CT10	FC115	Adv. cond. ind. by 33g in op. code. (oper in "u" 6l___)	
		42	RA	RA7	FC4	Adv. Nrp by 2 in "u" and "v"
70	43	MJ	0	IS2		
	44	TP	WT3	A	Operation symbol to "A"	
	45	EJ	FC117	IS50	Symbol = 16000 (POW ½)?	
	46	EJ	FC120	IS50	Symbol = 16100 (POW -½)?	
	47	MJ	0	SS		
	50	TP	CT10	A	Condition indicator to A	
	51	EJ	FC10	IS54	Cond. ind = 4?	

71

52	RA	CT1	FC4	No - advance Crpt by 2 in "u" and "v"
53	MJ	0	SS	
54	RA	CT1	FC3	Advance Crpt by 1 in "u" and "v"
55	RA	CT2	FC3	Advance Crct by 1 in "u" and "v"
56	MJ	0	SS	
	CA	IS57		

		IA	FD				
floating divide	72	0	TP	RC3	WT5		
		1	RJ	VC50	VC		
		2	QJ	FD3	FD5		
		3	RJ	FD26	FD22		
		4	MJ	0	FD14		
		5	RJ	FD34	FD31		
floating subtract	73	6	MJ	0	FD20		
		7	TP	RC2	WT5		
		10	RJ	VC50	VC		
		11	QJ	FD12	FD16		
		12	RJ	FD26	FD22		
		13	TV	RC11	PN61		
floating subtract	74	14	TP	FC10	CT10		
		15	MJ	0	RR43		
		16	RJ	FD34	FD31		
		17	RA	RA7	FC3		
		20	RA	CT10	FC10		
		no subscript word	K1	21	MJ	0	RR43
22	TV			RC10	PN61		
23	RJ			RR22	RR		
24	TV			WT5	WT1		
25	TP			RA6	A		
26	EJ			WT1	[30000]		
27	SP			RA6	17		
30	MJ			0	FP7		
no subscript word	L2			31	RJ	RR42	RR25
				32	TV	WT5	WT1
		33	TP	RA6	A		
		34	EJ	WT1	[30000]		
		35	RA	RA7	CT7		
		36	MJ	0	RR44		
		CA	FD37				

Floating Divide and Floating Subtract Operators

Dummy fl. divide inst. → working temp.
 Check variables and set switch (H)
 Is there a subscript word?
 Subscript word

No subscript word

Dummy fl. subtract inst. → working temp.
 Check variables and set switch (H)
 Is there a subscript word?
 Subscript word

Set switch (M) → (M2)
 Set cond. ind. → 4 in op. code (oper. for "v" in Q)

No subscript word

Advance Nrp by one in "u" and "v"
 Set cond. ind. → 16 in op. code (oper. for "v" in Q)

Set switch (M) → (M1)
 Jump to redundancy routine
 "v" of dummy inst. → "v" of working temp.
 P.R. counter → A

P.R. counter → "u" of A

Jump to Redundancy Routine
 "v" of dummy inst → "v" of working temp.
 P.R. counter → A
 P.R. counter = "v" of dummy inst.
 Advance Nrp by increment (I)

					Floating Plus and Floating Multiply Operators
75	0	IA TP	FP RC21	WT3	Dummy fl. plus w/zero in "u" and "v" → working temp.
	1	MJ	0	FP3	
76	2	TP	RC22	WT3	Dummy fl. mult. w/zero in "u" and "v" → working temp.
77	3	RJ	VS64	VS	Sort operands and set switch (H)
	4	RA	WT5	WT3	Dummy floating [multiply] with operands → "A" and temp.
	5	QJ	FP6	FP26	Is there a subscript word? yes to FP6
78	6	RJ	RR22	RR	Is operation redundant? no to FP7
78A	7	TU	WT6	WT	"u" of s.s. word → working temp.
	10	EJ	WT	FP23	Is P.R. counter = "u" of s.s. word (s.s. for "u" in A)
	11	TP	RA6	A	P.R. counter → "v" of A
	12	TV	WT6	WT1	"v" of s.s. word → working temp.
	13	EJ	WT1	FP17	P.R. counter = "v" of s.s. word (s.s. for "v" in A)
	14	RA	RA7	CT7	Advance Nrp by increment (I)
	15	TP	FC	CT10	Set cond. ind. → Zero (neither s.s. in A)
	16	MJ	0	RR44	
79	17	TV	RC3	PN14	Set switch (G) → (G2)
	20	TV	RC5	PN44	Set switch (J) → (J2)
	21	TP	FC6	CT10	Set cond. ind. → 2 in op. code (s.s. for "v" in A)
	22	MJ	0	FP24	
80	23	TP	FC5	CT10	Set cond. ind. → 1 in op. code (s.s. for "u" in A)
	24	RJ	LA	LA6	Enter P.R. in "A" list
	25	MJ	0	RR44	
81	26	RJ	RR42	RR25	Is operation redundant? no to FP27
	27	RA	RA7	CT7	Advance Nrp by increment (I)
	30	MJ	0	RR44	
		CA	FP31		

82	0	IA PO RJ ES	ES12	Fixed Point Plus Operator Advance "D" to available dummy inst. address
	1	TV A	P04	Available dummy inst. address → "v" of AT
	2	TV A	WT4	Address of dummy inst. → "v" of temp.
	3	RJ OS13	OS	Sorted operands → "u" and "v" of A
	4	AT RC23	[30000]	Dummy "fixed plus" inst. w/operands → Dummy List
	5	TU A	WT	Operand in "u" of dummy inst. → working temp.
	6	RJ ES	ES1	Search Expanded List for redundancy
	7	SJ P010	P021	"u" if inst. not redundant - "v" if inst. is redundant
83	10	SP RA6	17	Partial result counter → "u" of A
	11	EJ WT	P016	Is P.R. counter = "u" of dummy inst. (op- erand in A?)
	12	RA RA7	FC4	No, advance #lines in running prog (Nrp) by 2 in "v"
83A	13	RJ SR25	SR4	P.R. value → Exp. List and Oper. List; cond. ind. → Expanded List
	14	TP FC	CT7	Set increment (I) → zero
84	15	MJ 0	SS	
	16	RA RA7	FC3	Advance #lines in running prog (Nrp) by 1 in "u" and "v"
84A	17	RJ SR25	SR7	P.R. value → "A" List, Exp. List and Oper. List; cond. ind. → Exp. List
82A	20	MJ 0	P014	
	21	TP RA4	RA5	Set D = γ (delete Dummy List from Expand- ed List)
	22	RJ RS	RS1	Redundant P.R. value → Expanded List and Red. P.R. List
	23	MJ 0 CA P024	SS	

85	0	IA MO		Fixed Point Multiply Operator
		RJ ES	ES12	Advance "D" to available dummy inst. address
86	1	TV A	M04	Available dummy inst. address → "v" of AT
	2	TV A	WT4	Address of dummy inst → "v" of temp.
	3	RJ OS13	OS	Sorted operands → "u" and "v" of A
	4	AT RC25	[30000]	Dummy "fixed mult" inst w/operands → Dummy List
87	5	TU A	WT	Operand in "u" of dummy inst → working temp.
	6	RJ ES	ES1	Search Expanded List for redundancy
	7	SJ M010	M021	"u" if not redundant; "v" if redundant
	10	SP RA6	17	Partial result counter → "u" of A
	11	EJ WT	M016	Is P.R. counter = "u" of dummy inst.? (operand in A)
	12	RJ SR25	SR4	No, P.R. symbol → Exp. List and Oper. List; cond. ind. → Exp. List
88	13	RA RA7	FC3	Advance #lines in running prog. (Nrp) by 1 in "v"
	14	TP FC3	CT7	Set increment (I) → one in "u" and "v"
	15	MJ 0	SS	
89	16	RA RA7	FC3	Advance #lines in running prog (Nrp) by 1 in "u" and "v"
	17	RJ SR25	SR7	P.R. value → "A" list, Exp. List and Oper. List; cond. ind → Exp. List
86A	20	MJ 0	M014	
	21	TP RA4	RA5	Set D = γ (delete Dummy List from Expanded List)
	22	RJ RS	RS1	Redundant P.R. → Expanded List and Red. P.R. List
	23	SJ M024	M026	Was P.R. previously entered in Redundant P.R. List?
	24	TP WT1	A	
	25	RJ LA	LA1	Redundant P.R. in "A" List (if yes, advance Nrp by 1)
	26	MJ 0	SS	
		CA M027		

90	0	IA NO		Fixed Point Subtract Operator	
		RJ ES	ES12	Advance "D" to available dummy inst. address	
	1	TV A	NO12	Dummy inst. address → "v" of AT	
	2	TV A	WT4	Dummy inst. address → "v" of working temp	
	3	RJ BR	BR1	Decrease address in Oper. List (β) by 1 in "u" and "v"	
	4	TU RA	NO5	Address of first operand → "u" of NI	
	5	TP [30000]	WT5	First operand → "v" of working temp.	
	6	RJ BR	BR1	Decrease add. in Oper. List (β) by 1 in "u" and "v"	
	7	TU RA	NO10	Address of second operand → "u" of NI	
	10	SP [30000]	17	Second operand → "u" of A	
	11	SA WT5	0	Operand → "u" and "v" of A	
	12	AT RC24	[30000]	Dummy "fixed minus" inst. w/operands → Dummy List	
91	13	TU A	WT	Operand in "u" of dummy inst. → working temp.	
	14	TV A	WT1	Operand in "v" of dummy inst. → working temp.	
	15	RJ ES	ES1	Search Expanded List for redundancy	
	16	SJ NO17	PO21	"u" if inst. not redundant - "v" if inst. redundant	
	17	SP RA6	17	Partial result counter → "u" of A	
	20	EJ WT	PO17	Is P.R. counter = "u" of dummy inst. (operand in A)	
	21	RA RA7	FC4	Advance #lines in running prog (Nrp) by 2 in "u" and "v"	
	22	TP RA6	A	Partial result counter → "v" of A	
	23	EJ WT1	NO26	Operand for "v" in A	
	24	RJ SR25	SR4	P.R. value → Exp. List and Oper. List; cond, ind → Exp. List	
	92	25	MJ 0	NO30	
		26	TP FC6	CT10	Set condition indicator → two (operand for "v" in A)
93	27	RJ SR25	SR10	P.R. value → "A" list, Exp. List and Oper. List; cond. ind. → Exp. List	
	30	TP FC	CT7	Set increment (I) → Zero	
	31	MJ 0	SS		
		CA NO32			

				Divide Operator
94	0	IA DO RJ ES	ES12	Advance "D" to available dummy inst. address
	1	TV A	DO12	Dummy inst. address → "v" of AT
	2	TV A	WT4	Dummy inst. address → "v" of working temp.
	3	RJ BR	BR1	Decrease add. in Oper. List (β) by 1 in "u" and "v"
	4	TU RA	DO5	Address of first operand → "u" of NI
	5	TP [30000]	WT5	First operand → "v" of dummy inst. (divisor)
	6	RJ BR	BR1	Decrease add. in Oper. List (β) by 1 in "u" and "v"
	7	TU RA	DO10	Address of second operand → "u" of NI
	10	SP [30000]	17	Second operand → "u" of A (dividend)
	11	SA WT5	0	Operand → "u" and "v" of A
	12	AT RC26	[30000]	Dummy "fixed divide" inst. w/operands → Dummy List
	13	TU A	WT	Operand in "u" of dummy inst → working temp.
	14	RJ ES	ES1	Search Expanded List for redundancy
	15	SJ DO16	MO21	"u" if inst. not redundant - "v" if inst. redundant
	16	SP RA6	17	Partial result counter → "u" of A
95	17	EJ WT	DO24	Is P.R. counter = "u" of dummy inst. (operand in A?)
	20	RA RA7	FC4	Advance #lines in running prog (Nrp) by 2 in "u" and "v"
	21	RJ SR25	SR4	P.R. value → Exp. List and Oper. List; cond. ind → Exp. List
	22	TP FC	CT7	Set increment (I) → Zero
	23	MJ 0	SS	
97	24	RA RA7	FC3	Advance #lines in running prog (Nrp) by 1 in "u" and "v"
	25	RJ SR25	SR7	P.R. value → "A" list, Exp. List and Oper List; cond. ind. → Exp. List
	26	MJ 0 CA DO27	DO22	

Floating Point Unary Minus
and Absolute Value

(98)	0	IA	FN		
	1	TP	RC5	WT5	Dummy floating Unary minus → temp 5
	2	MJ	O	FN3	
(99)	2	TP	RC6	WT5	Dummy floating absolute value → temp 5
(100)	3	RJ	VC50	VC5	Check variable and set switch (H)
	4	QJ	FN5	FN60	Is there a subscript? no, take "v"
	5	TU	RA	FN6	Yes
(101)	6	TV	[30000]	WT5	Subscript → "v" of dummy instruction
	7	TP	WT5	A	Dummy instruction to "A"
	10	RJ	ES	ES1	Search Exp. List for instruction
	11	SJ	FN12	FN26	Is instruction redundant? yes to FN26
	12	RJ	ES	ES12	Advance dummy tally by one.
	13	TV	RA5	FN14	Available address in Exp. List → "v" of TP
	14	TP	WT5	[30000]	Inst. at D → Expanded List
	15	TV	WT5	WT1	s.s. → "v" of working temp.
	16	TP	RA6	A	P.R. counter → A
	17	EJ	WT1	FN23	Is P.R. counter = subscript? (s.s. for "u" in "A"?)
(102)	20	TP	FC	CT10	Set cond. ind. → Zero (neither s.s. in "A")
	21	RA	RA7	CT7	Advance Nrp by increment (I)
	22	MJ	O	FN102	
(103)	23	TP	FC5	CT10	Set cond. ind. → one (s.s. for "u" in A)
	24	RJ	LA	LA6	Enter P.R. value in "A" List
	25	MJ	O	FN102	
(104)	26	RJ	RS	RS1	Redundant P.R. value → Oper. List and Red. P.R. List
	27	SJ	FN30	SS	Was redundant P.R. in Red. P.R. List?
	30	TP	WT1	A	Redundant P.R. → A
	31	RJ	LQ	LQ17	Is redundant P.R. in "Q" List (yes → NI; no → SS)
	32	SN	Q	17	-jnr → "u" of A
	33	SA	RA2	0	+r → "u" of A
	34	SS	FC2	0	(r-1) → "u" of A
	35	SA	LQ21	25	IQ+r-1 → "v" of AL (address of P.R. in "Q" List)
	36	LT	O	A	IQ+r-1 → "v" of AR
	37	TV	A	FN40	Address of redundant P.R. in "Q" List → "v" of NI
	40	TP	FC	[30000]	Delete redundant P.R. from "Q" List
	41	TU	RS2	FN43	Add. of redundant P.R. in Exp. List → "u" of TP
	42	RA	FN43	FC2	Adv. to address after redundant P.R. in Exp. List
	43	TP	[30000]	Q	Mask for op. code and "v" → Q
	44	QT	FC37	WT2	Op. code and "v" of word following Red. P.R. → working temp.

	45	TP	WT2	A	Op. code and "v" of word following red. P.R.→ A
	46	TP	RC2	WT2	Dummy FS→ working temp.
	47	TV	WT1	WT2	Redundant P.R.→ "v" of dummy FS in working temp.
	50	EJ	WT2	FN52	Is inst. following red. P.R. in Exp. List = FS with red. P.R. in "v"?
105	51	MJ	0	SS	
	52	TU	Q	WT	"u" of inst. following red. P.R.→ "u" of working temp.
	53	TP	FC56	A	76777→ "u" of A
	54	TJ	WT	FN107	"u" of FS inst. 77... type? no to FN55
	55	TP	FC54	A	73777→ "u" of A
	56	TJ	WT	FN110	"u" of FS inst. 75... or 76...type?
108	57	MJ	0	FN107	
	60	TV	FC	WT5	Zero to "v" of dummy instruction
	61	TP	WT5	A	Dummy instruction→ A
	62	RJ	ES	ES1	Search Exp. List for instruction
	63	SJ	FN64	FN26	Is instruction redundant? yes, take "v"
	64	RJ	ES	ES12	Advance dummy tally by one
	65	TV	RA5	FN66	Available address in Expanded List→ "v" of TP
	66	TP	WT5	[30000]	Instruction→ Expanded List
	67	RA	RA7	FC3	Advance Nrp by one
	70	TP	FC13	CT10	Set cond. ind.→ 12 in op. code (neither "u" nor "v" subs)
	71	SP	RA6	17	P.R. counter→ "u" of A
	72	TU	WT5	WT	"u" of dummy inst.→ "u" of working temp.
	73	EJ	WT	FN77	P.R. counter = "u" of dummy inst.?
109	74	RA	RA7	CT7	Advance Nrp by increment (I)
	75	RJ	EK25	EK6	To 61... routine "u" ent.
	76	MJ	0	FN101	
110	77	RA	CT10	FC7	Set cond. ind.→ 15 in op. code
	100	RJ	LQ	LQ7	Enter partial result symbol in "Q" List
111	101	TV	RC7	PN	Set switch (H) to (1)
111A	102	TV	RA5	WT4	Address of dummy inst.→ "v" of working temp.
	103	TP	FC	CT7	Set increment (I)→ Zero
	104	TV	RC20	PN61	Set switch (M) to (M3)
	105	RJ	SR25	SR11	P.R. value→ Oper. List (β) and Exp. List (γ); cond. ind→ Exp. List
106	106	MJ	0	PN	
107	107	RS	RA7	FC3	Reduce Nrp by one
	110	TU	FN43	FN112	
	111	RA	FN112	FC2	
	112	RS	[30000]	FC10	Change ind for "FS"→ operand for "v" not in Q
	113	MJ	0	SS	
		CA	FN114		

Fixed Point Unary Minus and
Absolute Value Operators

(112)	0	IA NF	WT5	Dummy fixed pt. unary minus inst. to temp
	1	MJ 0	NF3	
(113)	2	TP RC30	WT5	Dummy fixed pt. abs. value inst. to temp
(114)	3	RJ ES	ES12	Adv. "D" to available dummy inst. address
	4	TV A	NF12	Preset address in Exp. List for dummy inst.
	5	TV A	WT4	Store address for dummy inst. in temp
	6	RJ BR	BR1	Decrease address in Operand List (β) by 1
	7	TU RA	NF10	Preset address of next operand
	10	TP [30000]	Q	Obtain next operand from Operand List
	11	SP Q	17	
	12	AT WT5	[30000]	Dummy instruction with operands to Dummy List
(115)	13	TP Q	WT1	
	14	RJ ES	ES1	Search Expanded List for redundancy
	15	SJ NF16	NF22	"u" if not redundant, "v" if redundant
	16	RA RA7	FC3	Advance Nrp by 1
	17	TP WT1	A	Operand to "A"
	20	EJ RA6	P017	P.R. counter = operand? (operand in "A"?)
	21	MJ 0	P013	No
(116)	22	TP RA4	RA5	Delete Dummy List from Expanded List (set D = γ)
	23	RJ RS	RS1	Was redundant P.R. in redundant P.R. List
	24	SJ NE	SS	"u" if no, "v" if yes
		CA NF25		

Fixed Point Unary Minus and
Absolute Value Operators

	0	IA NE	A	Redundant partial result to A
	1	EJ RA6	NE5	Redundant P.R. = P.R. counter (current P.R.)?
	2	TU RA3	NE3	
	3	RP [30000]	SS	Search "A" List
	4	EJ XA	NE6	Is redundant P.R. in "A" List?
	5	TP FC122	CT7	Set increment (I) to minus one
	6	RA RA7	FC3	Advance Nrp by one
	7	MJ 0	SS	
		CA NE10		

	IA	EE		Storage Operator (space-period)	
117	0	TP	RC20	WT5	Dummy store inst. to working temp 5
	1	RJ	BR	BR1	Decrease β by 1 in "u" and "v"
	2	TU	RA	EE3	Preset address of operand
	3	SP	[30000]	17	Operand to "u" of A
	4	TU	A	WT5	Operand to "u" of working temp 5
	5	TU	A	WT	Operand to "u" of working temp
	6	TP	FC54	A	74777 to "u" of A
	7	TJ	WT	EG	Operand > 74777? (operand subscripted?)
	10	RJ	BR	BR1	Decrease β by 1 in "u" and "v"
	11	TU	RA	EE12	Address of 2nd operand \rightarrow "u" of NI
	12	TV	[30000]	WT5	Operand \rightarrow "u" of working temp 5
	13	TV	WT5	WT1	Operand \rightarrow "v" of working temp 1
	14	TP	FC52	A	76777 \rightarrow "v" of A
118	15	TJ	WT1	EF26	Operand in "v" > 76777? (i.e. 77--- type)
	16	TP	FC76	A	74777 \rightarrow "v" of A
	17	TJ	WT1	EF5	Operand in "v" > 74777? (i.e. 75--- type)
	20	TP	FC13	CT10	Set cond. ind. \rightarrow ⑫ in op. code - ("u" and "v" non-subs)
	21	RA	RA7	FC4	Advance Nrp by 2 in "u" and "v"
119	22	TU	WT5	WT	Operand for "u" to temp
	23	SP	RA6	17	P.R. counter \rightarrow "u" of A
	24	EJ	WT	EF	P.R. counter = operand? (i.e. oper. for "u" in "Q"?)
	25	RJ	EK25	EK	No, to 61--- routine ("u" and "v" ent.)
120	26	TP	FC36	Q	Mask for op. code \rightarrow Q
	27	QS	CT10	WT6	Condition indicator to op. code temp 6
	30	RJ	ES	ES12	Advance D by 1 \rightarrow next available add. in Exp. List
	31	TV	A	EE32	Next available add. in Exp. List \rightarrow "v" of NI
	32	TP	WT5	[30000]	Dummy storage instruction \rightarrow Expanded List
121	33	RJ	ES	ES12	Advance D by 1 \rightarrow next available add. in Exp. List
	34	TV	A	EE35	Next available address in Exp. List \rightarrow "v" of NI
	35	TP	WT6	[30000]	Indicator and s.s. word \rightarrow Expanded List
	36	MJ	O	ER	Exit-to end redundancy phase
		CA	EE37		

		IA	EF		Storage Operator (continued)
122	0	RJ	LA	LA6	Store P.R. value in "A" list
	1	RJ	EK25	EK14	To 6l___routine ("v" ent.)
123	2	RA	CT10	FC7	Adv. cond. ind. by 3 (oper. for "u" in Q)
	3	RJ	LQ	LQ7	Store P.R. value in "Q" List
	4	MJ	O	EE26	
124	5	RJ	BR	BR1	Decrease β by 1 in "u" and "v"
	6	TU	RA	EF7	Address of s.s. for "v" \rightarrow "u" of NI
	7	TV	[30000]	WT6	s.s. at $\beta \rightarrow$ "v" of temp 6
	10	TP	FC62	CT10	Set cond. ind. \rightarrow ⑪ in op. code ("u" non-subs and "v" 75___)
	11	RA	RA7	FC25	Advance Nrp by 4 in "u" and "v"
	12	TV	WT6	WT1	Subscript for "v" operand to temp 1
	13	TP	RA6	A	P.R. value \rightarrow "v" of A
	14	EJ	WT1	EF22	P.R. counter = subscript? (s.s. for "v" in "A"?)
125	15	TU	WT5	WT	"u" operand to temp 0
	16	SP	RA6	17	P.R. value \rightarrow "u" of A
	17	EJ	WT	EF2	"u" operand = P.R. counter? (oper. for "u" in "Q"?)
	20	RJ	EK25	EK6	To 6l___routine ("u" ent.)
	21	MJ	O	EE26	
126	22	RJ	EK25	EK6	To 6l___routine ("u" ent.)
126A	23	RA	CT10	FC6	Adv. cond. ind. by 2 in op. code
	24	RJ	LA	LA6	Store P.R. value in "A" list
	25	MJ	O	EE26	
127	26	RJ	BR	BR1	Decrease β by 1 in "u" and "v"
	27	TU	RA	EF30	Address of s.s. for "v" \rightarrow "u" of NI
	30	TV	[30000]	WT6	s.s. at $\beta \rightarrow$ "v" of temp 6
	31	TP	FC11	CT10	Set cond. ind. \rightarrow ⑤ ("u" non-subs and "v" 77___)
	32	RA	CT	FC3	Advance #rel. const. (Crc) by 1 in "u" and "v"
	33	TV	WT6	WT1	Subscript for "v" operand to temp 1
	34	TP	RA6	A	P.R. value \rightarrow "v" of A
	35	EJ	WT1	EF40	Subscript = P.R. counter? (s.s. for "v" in "A"?)
	36	RA	RA7	FC25	Advance Nrp by 4 in "u" and "v"
	37	MJ	O	EF15	
	40	RA	RA7	FC24	Advance Nrp by 3 in "u" and "v"
	41	MJ	O	EF22	
		CA	EF42		

				Storage Operator (continued)
	IA	EG		
128	0	RJ BR	BR1	Decrease β by 1 in "u" and "v"
	1	TU RA	EG2	Address of s.s. \rightarrow "u" of NI
	2	SP [30000]	17	s.s. \rightarrow "u" of A
	3	TP A	WT6	s.s. \rightarrow "u" of temp 6
	4	RJ BR	BR1	Decrease β by 1 in "u" and "v"
	5	TU RA	EG6	Add. of oper. for "v" \rightarrow "u" of NI
	6	TV [30000]	WT5	Operand \rightarrow "v" of temp 5
	7	TV WT5	WT1	"v" operand to "v" of working temp 1
	10	TP FC52	A	76777 \rightarrow "v" of A
129	11	TJ WT1	EH23	Operand for "v" > 76777? (i.e. 77___type)
	12	TP FC76	A	74777 \rightarrow "v" of A
	13	TJ WT1	EH	"v" operand > 74777? (i.e. 76___type)
	14	TU WT5	WT	"u" operand to temp 0
	15	TP FC56	A	76777 \rightarrow "u" of A
	16	TJ WT	EG22	"u" operand > 76777? (i.e. 77___type)
	17	TP FC	CT10	Zero to cond. ind. ("u" 75___or 76___ and "v" non-subs)
	20	RA RA7	FC26	Adv. Nrp by 5 in "u" and "v"
	21	MJ 0	EG25	
130	22	TP FC6	CT10	Set cond. Ind. \rightarrow ② in op. code ("u" 77___ and "v" non-subs.)
	23	RA RA7	FC25	Adv. Nrp by 4 in "u" and "v"
	24	RA CT	FC3	Adv. Crc by 1
131	25	RJ EK25	EK14	To 61___routine ("v" ent.)
	26	TU WT6	WT	Subscript for "u" operand to temp 0
	27	SP RA6	17	P.R. counter \rightarrow A
	30	EJ WT	EG32	Subscript = P.R. counter? (s.s. for "u" in "A"?)
	31	MJ 0	EE26	
132	32	RA CT10	FC5	Adv. cond. ind by 1 in op. code (s.s. for "u" in A)
	33	MJ 0	EF24	
		CA	EG34	

				Storage Operator (continued)
(133)	0	IA EH		
	1	RJ BR	BR1	Decrease β by 1 in "u" and "v"
	2	TU RA	EH2	Address of s.s. \rightarrow "u" of NI
(134)	3	TV [30000]	WT6	s.s. \rightarrow "v" of temp 6
	4	TU WT5	WT	"u" operand to "u" of temp 0
	5	TP FC56	A	76777 \rightarrow "u" of A
	6	TJ WT	EH11	"u" operand > 76777? (i.e. 77___type)
	7	TP FC16	CT10	Set cond. ind. to 22 ("u" 75___or 76___ and "v" 75___)
	10	RA RA7	FC121	Adv. Nrp by 7 in "u" and "v"
(135)	11	MJ 0	EH14	
	12	TP FC20	CT10	Set cond. ind. \rightarrow (30) in op. code ("u" 77___ and "v" 75___)
	13	RA RA7	FC27	Adv. Nrp by 6 in "u" and "v"
(136)	14	RA CT	FC3	Adv. Crc by 1 in "u" and "v"
	15	TU WT6	WT	Subscript for "u" to working temp
	16	SP RA6	17	P.R. counter \rightarrow "u" of A
	17	EJ WT	EG32	Subscript = P.R. counter? (s.s. for "u" in "A"?)
(137)	20	TV WT6	WT1	Subscript for "v" to working temp
	21	TP RA6	A	P.R. counter \rightarrow "v" of A
	22	EJ WT1	EF23	Subscript = P.R. counter? (s.s. for "v" in "A"?)
(138)	23	MJ 0	EE26	
	24	RJ BR	BR1	Decrease β by 1 in "u" and "v"
	25	TU RA	EH25	Add. of s.s. \rightarrow "u" of NI
	26	TV [30000]	WT6	Subscript to "v" of temp 6
	27	TU WT5	WT	"u" operand to working temp
	30	TP FC56	A	76777 \rightarrow "u" of A
	31	TJ WT	EH34	"u" operand > 76777? (i.e. 77___type)
	32	TP FC17	CT10	Set cond. ind. to 25 ("u" 75___or 76___ and "v" 77___)
	33	RA RA7	FC27	Adv. Nrp by 6 in "u" and "v"
(139)	34	MJ 0	EH36	
	35	TP FC14	CT10	Set cond. ind. \rightarrow (17) in op. code ("u" and "v" 77___)
(140)	36	RA RA7	FC26	Adv. Nrp by 5 in "u" and "v"
	37	RA CT	FC1	Adv. Crc by 1 in "u" and "v"
		MJ 0	EH14	
		CA EH40		

	IA	VC		Subroutine to Check Variables
141	0	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	1	TU	RA VC3	Address of 1st operand \rightarrow "u" of TV
	2	TP	FC76 A	74777 \rightarrow "v" of A
	3	TV	[30000] WT1	Operand \rightarrow "v" of working temp
141A	4	TJ	WT1 VC24	1st operand > 74777? (i.e. subscripted?)
	5	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	6	TU	RA VC7	Address of second operand \rightarrow "u" of NI
	7	SP	[30000] 17	2nd operand \rightarrow "u" of A
	10	TU	A WT	2nd operand \rightarrow "u" of working temp
	11	TU	A WT5	2nd operand \rightarrow "u" of temp 5
	12	TP	FC54 A	74777 \rightarrow "u" of A
	13	TJ	WT VC16	2nd operand > 74777? (i.e. subscripted?)
	14	TP	FC Q	(Q ₃₅ = 0) no subscript word
	15	MJ	O VC46	
142	16	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	17	TU	RA VC20	Address of s.s. for oper. in "u" \rightarrow "u" of NI
	20	SP	[30000] 17	s.s. for oper. in "u" \rightarrow "u" of A
	21	TP	A WT6	s.s. \rightarrow "u" of temp. 6
	22	TV	RC6 PN	Set switch (H) to (H3), "u" subs and "v" non-subs
	23	MJ	O VC45	
143	24	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	25	TU	RA VC26	Address of s.s. for oper. in "v" \rightarrow "u" of NI
	26	TP	[30000] WT6	s.s. \rightarrow "v" of temp 6
	27	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	30	TU	RA VC31	Address of 2nd operand \rightarrow "u" of NI
	31	SP	[30000] 17	2nd operand \rightarrow "u" of A
	32	TU	A WT	2nd operand \rightarrow "u" of working temp
	33	TU	A WT5	2nd operand \rightarrow "u" of temp 5
	34	TP	FC54 A	74777 \rightarrow "u" of A
	35	TJ	WT VC40	2nd operand > 74777?
	36	TV	RC PN	Set switch (H) to (H2), "u" non-subs and "v" subs
	37	MJ	O VC45	
144	40	RJ	BR BR1	Decrease β by 1 in "u" and "v"
	41	TU	RA VC42	Address of s.s. for oper. in "u" \rightarrow "u" of NI
	42	SP	[30000] 17	s.s. for oper. in "u" \rightarrow "u" of A
	43	TU	A WT6	s.s. \rightarrow "u" of temp 6
	44	TV	RC1 PN	Set switch (H) to (H1), "u" non-subs and "v" subs
145	45	TP	FC36 Q	(Q ₃₅ = 1) subscript word
146	46	TV	WT1 WT5	Operand \rightarrow "v" of dummy inst.
	47	TP	WT5 A	Dummy inst. w/operands \rightarrow A
	50	MJ	O [30000]	Exit
		CA	VC51	

					Subroutine to Sort Operands for Floating Plus or Multiply
(147)	0	IA VS			
	1	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	2	TU RA	VS2		Address of 1st operand \rightarrow "u" of NI
	3	TP [30000]	Q		First operand \rightarrow Q
	4	TP FC76	A		74777 \rightarrow "v" of A
	5	TJ Q	VS31		First operand > 74777? (i.e. subscripted)
	6	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	7	TU RA	VS7		Address of 2nd operand \rightarrow "u" of NI
	10	TP [30000]	A		Second operand \rightarrow A
(148)	10	TJ Q	VS25		First operand > second operand?
	11	LQ Q	17		First operand \rightarrow "u" of Q
	12	AT Q	WT5		Operands \rightarrow "u" and "v" of temp 5
	13	TV WT5	WT1		Second operand \rightarrow "v" of working temp
	14	TP FC76	A		74777 \rightarrow A
(150)	15	TJ WT1	VS20		Second operand > 74777? (i.e. subscripted)
	16	TP FC	Q		(Q ₃₅ = 0) no subscript word
	17	MJ O	VS64		
	20	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	21	TU RA	VS22		Address of s.s. for oper. in "v" \rightarrow "u" of NI
(148A)	22	TP [30000]	WT6		s.s. for oper. in "v" \rightarrow "v" of temp 6
	23	TV RC	PN		Set (H) to (H ₂) "u" non-sub and "v" subscripted
(149)	24	MJ O	VS63		
	25	TV A	WT1		Second operand \rightarrow "v" of working temp
	26	LA A	17		Second operand \rightarrow "u" of A
	27	AT Q	WT5		Operands \rightarrow "u" and "v" of temp 5
(151)	30	MJ O	VS16		
	31	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	32	TU RA	VS33		Address of s.s. for first oper. \rightarrow "u" of NI
(152)	33	TP [30000]	WT6		s.s. \rightarrow "v" of temp 6
	34	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	35	TU RA	VS36		Address of second operand \rightarrow "u" of NI
	36	TP [30000]	A		Second operand \rightarrow A
	37	TJ Q	VS50		First operand > second operand?
	40	LQ Q	17		First operand \rightarrow "u" of Q
	41	AT Q	WT5		Operands \rightarrow "u" and "v" of temp 5
	42	RJ BR	BR1		Decrease β by 1 in "u" and "v"
	43	TU RA	VS45		Address of second s.s. \rightarrow "u" of TV
(153)	44	LA WT6	17		First s.s. \rightarrow "u" of temp 6
	45	TV [30000]	WT6		Second s.s. \rightarrow "v" of temp 6
	46	TV RC1	PN		Set (H) to (H ₁) "u" and "v" subscripted
(154)	47	MJ O	VS63		
	50	TV A	WT1		Second oper. \rightarrow "v" of working temp
	51	LA A	17		Second operand \rightarrow "u" of A
	52	AT Q	WT5		Operands \rightarrow "u" and "v" of temp 5
	53	TP FC76	A		74777 \rightarrow "v" of A

	54	TJ	WT1	VS56	Oper. in "u" (2nd oper.) > 74777? (i.e. subscripted)
155	55	MJ	0	VS23	
	56	RJ	BR	BR1	Decrease β by 1 in "u" and "v"
	57	TU	RA	VS60	Address of s.s. for oper. in "u" \rightarrow "u" of NI
	60	SP	[30000]	17	s.s. for 2nd oper. \rightarrow "u" of A
	61	TU	A	WT6	s.s. for 2nd oper. \rightarrow "u" of temp 6
156	62	TV	RC1	PN	Set \textcircled{H} to $\textcircled{H1}$ "u" and "v" subscripted
157	63	TP	FC36	Q	($Q_{35} = 1$) subscript word
	64	MJ	0	[Exit]	
		CA	VS65		

Set Condition Indicator for
Floating Point Operations

	IA	PN		
	0	MJ 0	[30000] Switch (H)	
(H1)	1	TU WT5	WT	
	2	TP FC56	A	76777 → "u" of A
	3	TJ WT	PN21	Is "u" of dummy inst. 77... ? yes; take "v"
	4	TV WT5	WT1	No
	5	TP FC52	A	76777 → "v" of A
	6	TJ WT1	PN12	Is "v" of dummy inst. 77... ? yes; take "v"
"u" 75--- "v" 75---	7	TP FC16	CT10	No; set condition indicator to 22 in op. code
	10	RA RA7	FC27	Advance Nrp by six in "u" and "v"
	11	MJ 0	PN66	
(158)	12	TP FC17	CT10	Set cond. ind. → 25 in op. code
	13	RA CT	FC3	Advance #rel. const (Crc) by 1 in "u" and "v"
"u" 75--- "v" 77---	14	MJ 0	[30000] Switch (G)	
(G1)	15	RA RA7	FC27	(G1) - advance Nrp by six in "u" and "v"
	16	MJ 0	PN66	
(G2)	17	RA RA7	FC26	(G2) - advance Nrp by five in "u" and "v"
(159)	20	MJ 0	PN66	
	21	TV WT5	WT1	Oper. in "v" of dummy inst. → working temp
	22	TP FC52	A	76777 → "v" of A
	23	TJ WT1	PN30	Is "v" of dummy inst 77... ? yes; take "v"
"u" 77--- "v" 75---	24	TP FC20	CT10	Set cond. ind. → 30 in op. code
	25	RA CT	FC3	Advance #rel. const. (Crc) by 1 in "u" and "v"
	26	RA RA7	FC26	Advance Nrp by six in "u" and "v"
	27	MJ 0	PN66	
(160)	30	TP FC14	CT10	Set cond. ind. → 17 in op. code
	31	RA CT	FC3	Advance #rel. const. (Crc) by 1 in "u" and "v"
"u" 77--- "v" 77---	32	RA RA7	FC25	Advance Nrp by four in "u" and "v"
	33	MJ 0	PN66	
(H2)	34	TV WT5	WT1	(H2) - oper. in "v" of dummy inst. to temp 1
"u" non-subs. "v" 75---	35	TP FC52	A	76777 → "v" of A
	36	TJ WT1	PN42	Is "v" of dummy inst. 77...? yes; take "v"
(161)	37	TP FC62	CT10	No; set condition indicator to 11 in op. code
	40	RA RA7	FC24	Advance Nrp by three in "u" and "v"
	41	MJ 0	PN72	
	42	TP FC11	CT10	Set cond. ind. → 5 in op. code
	43	RA CT	FC3	Advance #rel. const. (Crc) by 1 in "u" and "v"

"u" non-subst. "v" 77_---	J1	44	MJ	0	[30000]	Switch (J)
		45	RA	RA7	FC24	Advance Nrp by 3 in "u" and "v"
		46	MJ	0	PN72	
"u" 75_--- "v" non-subst.	J2	47	RA	RA7	FC4	Advance Nrp by 2 in "u" and "v"
		50	MJ	0	PN72	
		51	TU	WT5	WT	(H3) - "u" operand to working temp
"u" 77_--- "v" non-subst.	H3	52	TP	FC56	A	76777 → "u" of A
		53	TJ	WT	PN57	Is "u" of dummy inst. 77_---?
		54	TP	FC	CT10	Set cond. ind. → zero in op. code
"u" 75_--- "v" non-subst.	162	55	RA	RA7	FC25	Advance Nrp by four in "u" and "v"
		56	MJ	0	PN65	
		57	TP	FC6	CT10	Set cond. ind. → 2 in op. code
"u" 77_--- "v" non-subst.	163	60	RA	CT	FC3	Advance #rel. const. (Crc) by 1 in "u" and "v"
		61	MJ	0	[30000]	Switch (M)
		62	RA	RA7	FC24	(M1) -advance Nrp by three in "u" and "v"
"u" 77_--- "v" non-subst.	M1	63	MJ	0	PN65	
		64	RA	RA7	FC25	(M2) -advance Nrp by four in "u" and "v"
		65	RJ	EK25	EK14	To 61_--- routine ("v" ent.)
"u" 77_--- "v" non-subst.	M2	66	SP	SR23	17	Address of word following dummy inst. "u" of A
		67	TU	A	PN70	Add. of word following dummy inst. → "u" of NI
		70	RA	[30000]	CT10	Advance indicator following dummy inst. by cond. ind.
"u" 77_--- "v" non-subst.	164	71	MJ	0	SS	
		72	RJ	EK25	EK6	To 61_--- routine ("u" ent.)
		73	MJ	0	PN66	
"u" 77_--- "v" non-subst.	165	74	RA	RA7	FC24	(M3) -advance Nrp by 3 in "u" and "v"
		75	MJ	0	PN66	
		75	CA	PN76		

				Subroutine to Check for Redundant Floating Point Operation	
(167)	0	IA	RR		
	1	RJ	ES	ES1	Search Expanded List for Dummy inst.
	2	SJ	RR6	RR2	Is instruction redundant? No; to RR6
		TU	A	RR4	Address of s.s. word in Expanded List
					→ "u" of QT
	3	TP	FC35	Q	
	4	QT	[30000]	A	Subscript word from Expanded List → A
	5	EJ	WT6	RR47	Is subscript word in temp 6 redundant?
(168)	6	RJ	ES	ES12	Yes; to RR47
	7	TV	A	RR10	No, advance dummy tally by one
					Available address in Exp. List → "v" of
					NI
(169)	10	TP	WT5	[30000]	Dummy instruction to Expanded List
	11	RJ	ES	ES12	Advance dummy tally by one
	12	TV	A	RR13	Available address in Exp. List → "v" of
					NI
	13	TP	WT6	[30000]	Subscript word to Expanded List
	14	TV	RR10	WT4	Address of dummy instruction to "v" of
					temp
	15	TV	RC3	PN14	Set switch (G) to (G2)
	16	TV	RC4	PN44	Set switch (J) to (J1)
	17	SP	RA6	17	P.R. counter → "u" of A
	20	TU	WT5	WT	"u" of dummy inst. → working temp.
	21	EJ	WT	RR23	Is P.R. counter = "u" operand? (oper. for
					"u" in Q)
	22	MJ	0	[30000]	Exit - subscript word
	23	TP	FC7	CT10	Set cond. ind. → 3 in op. code (oper. for
					"u" in Q)
(170)	24	MJ	0	RR43	
	25	RJ	ES	ES1	Search Expanded List for dummy instruction
	26	SJ	RR27	RR50	Is instruction redundant? yes; to RR50
	27	RJ	ES	ES12	No, advance dummy tally by one
	30	TV	RA5	RR32	Available address in Exp. List → "v" of
					TP
		31	TV	RA5	WT4
	32	TP	WT5	[30000]	Dummy instruction to Expanded List
	33	RA	RA7	FC3	Advance Nrp by one in "u" and "v"
	34	TP	FC13	CT10	Set cond. ind. → 12 in op. code (neither
					"u" nor "v" subs)
(171)	35	SP	RA6	17	P.R. counter → "u" of A
	36	TU	WT5	WT	"u" of dummy inst. to working temp
	37	TV	RC7	PN	Set switch (H) → (1)
	40	EJ	WT	RR55	P.R. counter = "u" operand? (oper. for
					"u" in "Q")
(172)	41	RJ	EK25	EK	To 61... routine ("u" and "v" ent.)
	42	MJ	0	[30000]	Exit - no subscript word
(173)	43	RJ	LQ	LQ7	Enter P.R. value in Q List
	44	TP	FC3	CT7	Set increment (I) → one in "u" and "v"

operand for
"u" in Q

redundant instruction

(174)

45	RJ	SR25	SR11	P.R. value \rightarrow Oper. List (β) and Exp. List γ ; cond. ind. \rightarrow Exp. List
46	MJ	0	PN	Go to \textcircled{H} (prediction routine)
47	RA	RR4	FC2	Address of redundant P.R. value \rightarrow "u" of A
50	RJ	RS	RS1	Redundant P.R. value \rightarrow Oper. List and Red. P.R. List
51	SJ	RR52	SS	Was redundant P.R. in List?
52	TP	WT1	A	Redundant P.R. \rightarrow A
53	RJ	LQ	LQ1	Was redundant P.R. in Q List?
54	MJ	0	SS	
55	RA	CT10	FC7	Adv. cond. ind. by 3 in op. code (oper. for "u" in Q-set ind \rightarrow $\textcircled{15}$) To 61... routine ("v" ent.)
56	RJ	EK25	EK14	
57	MJ	0	RR43	
	CA	RR60		

Subroutine to Store Redundant
Partial Result

	IA	RS		
175	0	MJ	0	[30000] Exit
	1	TU	A	RS2 Address of redundant P.R. value → "u" of NI
	2	TV	[30000]	WT1 Redundant P.R. → "v" of working temp
	3	TV	RA	RS4 Available address in Operand List → "v" of NI
	4	TP	WT1	[30000] Redundant P.R. value → Operand List (β)
	5	RJ	BR	BR4 Advance address in Operand List (β) by 1 in "u" and "v"
	6	TP	WT1	A Redundant P.R. → A
	7	TU	RA10	RS10 Length of redundant P.R. List → jn of repeat
	10	RP	[30000]	RS12 Search Redundant P.R. List
176	11	EJ	RL	RS24 Is P.R. in "A" in Redundant P.R. List?
	12	TV	RA10	RS13 No; available address in Redundant P.R. List → "v" of NI
	13	TP	A	[30000] Redundant P.R. → redundant P.R. List
	14	RA	RA10	FC3 Advance available add. and jn for Red. P.R. List by one
	15	TJ	LV1	RS22 Redundant P.R. List too long?
	16	RJ	WA	WA1 Sent. # → print out
	17	TP	TO	UP3 Codeword → alarm print
	20	RJ	UP2	UP Alarm-Red. P.R. List too long [type: SENTENCE TOO LONG.]
	21	MJ	0	BQ6 Rewind tapes
	22	TP	FC36	A Set (A) → Red. P.R. was not in list (A-)
176A	23	MJ	0	RS
	24	TP	FC	A Set (A) → Red. P.R. was in list (A+)
	25	MJ	0	RS
		CA	RS26	

Subroutine to Sort Operands for
Fixed Plus or Multiply

177	0	RJ	BR	BR1 Decrease address in Operand List (β) by 1 in "u" and "v"
	1	TU	RA	OS4 Address of first operand → TP
	2	RJ	BR	BR1 Decrease address in Operand List (β) by 1 in "u" and "v"
	3	TU	RA	OS5 Address of second operand → TP
	4	TP	[30000]	A First operand → A
	5	TP	[30000]	Q Second operand → Q
	6	TJ	Q	OS11 Operand at Q > operand at A
	7	LQ	Q	17 Shift operand in Q → "u" of Q
	10	MJ	0	OS12
	11	LA	A	17 Shift operand in A → "u" of A
	12	SA	Q	0 Operands → "u" and "v" of A
	13	MJ	0	[30000]
		CA	OS14	

178

Subroutine to Store Callword in
Op. File 1

0	IA	FS		
	MJ	0	[30000]	Exit
1	SP	A	17	Callword → "u" of A
2	TU	RA1	FS3	Length Op. File 1 → jn of repeat
3	RP	[30000]	FS5	Search Op. File 1 for callword
4	EJ	FL2	FS	Callword in Op. File 1? no to FS5
5	TV	RA1	FS6	Available address in Op. File 1 → "v" of NI
6	TP	A	[30000]	Store callword in Op. File 1
7	RA	FL	FC1	Adv. #lines this Op. File 1 item by one
10	RA	RA1	FC3	Adv. available add. and jn for Op. File 1 by one
11	TJ	LV2	FS	Op. File 1 too long?
12	RJ	WA	WA1	Sent.# → print out
13	TP	TO	UP3	Codeword → alarm print
14	RJ	UP2	UP	Alarm-Op. File 1 too long [type: SEN- TENCE TOO LONG.]
15	MJ	0	BQ6	Rewind tapes
	CA	FS16		

179

179A

180

Subroutine
to Advance or Decrease Available Address
in Operand List (Beta Routine)

	IA	BR		
	MJ	0	[30000]	Exit
	RS	RA	FC3	Decrease address in Operand List (β) by 1 in "u" and "v"
	TJ	IA	EP4	Init. add. Oper. List > current add.? Yes → alarm #4
	MJ	0	BR	No; to exit
	RA	RA	FC3	Adv. address in Oper. List (β) by 1 in "u" and "v"
	TJ	LV	BR	Max. address in Oper. List > current add- ress?
	RJ	WA	WA1	Yes; type sentence number
	TP	TO	UP3	Code word → alarm print
	RJ	UP2	UP	Alarm-too many operands [type: SENTENCE TOO LONG.]
	MJ	0	BQ6	Rewind tapes and stop
	CA	BR12		

				Subroutine to Search for or Store Partial Result Symbol in "Q" List	
Search "Q" List (fl. plus, minus, mult., div.)	181	0	IA LQ		
		1	MJ 0	[30000] Exit	
		2	TU RA2	LQ2 Ent. - for search "Q" List (fl. plus, div., minus, mult.)	
		3	RP [30000]	LQ4 Search "Q" List	
		4	EJ XQ	LQ5 Is redundant P.R. in "Q" List	
		5	MJ 0	LQ22	
Store in "Q" List	182	6	RA RA7	FC3 Advance Nrp by one in "u" and "v"	
		7	MJ 0	LQ	
		10	TV RA2	LQ10 Ent. - for store in "Q" List	
		11	TP RA6	[30000] Enter P.R. in "Q" List	
		12	RA RA2	FC3 Adv. jn and add. in "Q" List by one in "u" and "v"	
		13	TJ LV3	LQ "Q" List too long	
		14	RJ WA	WA1 Sent. # → print out	
		15	TP TO	UP3 Codeword → alarm print	
		16	RJ UP2	UP Alarm - "Q" List too long [type: SENTENCE TOO LONG.]	
		Search "Q" List (fl. neg. & abs. val)	183	17	MJ 0
20	TU RA2			LQ20 Ent. - for search "Q" List (fl. neg and abs. val.)	
21	RP [30000]			SS Search "Q" List (exit to sym. search if Red. P.R. not in "Q" List)	
22	EJ XQ			LQ Is red. P.R. in "Q" List (return exit - red. P.R. in "Q" List)	
23	EJ RA6			LQ24 Redundant P.R. = P.R. counter?	
24	MJ 0			SS	
25	RA RA7			FC3 Adv. Nrp by one in "u" and "v"	
26	TP FC			CT7 Set increment (I) to zero	
				MJ 0	SS
				CA LQ27	

search "A" list

184

	IA	LA		
0	MJ	0	[30000]	Exit
1	TU	RA3	LA2	Ent.-for search "A" List
2	RP	[30000]	LA16	Search "A" List
3	EJ	XA	LA4	Is redundant P.R. in "A" List?
4	RA	RA7	FC3	Advance Nrp by one in "u" and "v"
5	MJ	0	LA	
6	TV	RA3	LA7	Ent.-for store in "A" List
7	TP	RA6	[30000]	Enter P.R. in "A" List
10	RA	RA3	FC3	Adv. jn and add. in "A" List by one in "u" and "v"
11	TJ	LV4	LA	"A" List too long?
12	RJ	WA	WA1	Sent. # - print out
13	TP	TO	UP3	Codeword → alarm print
14	RJ	UP2	UP	Alarm-"A" List too long [type: SENTENCE TOO LONG.]
15	MJ	0	BQ6	Rewind tapes and stop
16	EJ	RA6	LA20	Redundant P.R. = P.R. counter?
17	MJ	0	LA	No
20	RA	RA7	FC3	Adv. Nrp by 1 in "u" and "v"
21	TP	FC	CT7	Set increment (I) to Zero
22	MJ	0	LA	
	CA	LA23		

Store in "A" list

185

Subroutine to Search for or Store Partial Result Symbol in "A" List

*Note: Sent. callword from sorted list → first word in Exp. List
 Sent. number from sorted list → second word in Exp. List

				Input-dummy inst. in A	
				Subroutine to Search for Dummy Instruc-	
				tion or Advance Dummy Tally in Exp. List	
Advance dummy tally (D)	Search Exp. List	(186)	0	IA ES	[30000] Exit
			1	MJ 0	ES2 jn for repeat → NI
			2	TU RA4	ES10 Search Expanded List
			3	RP [30000]	ES4 Dummy inst = prev. entry in Exp. List?
			4	EJ EL2	ES4 yes; to ES4
			5	SN Q	17 -jn+r → "u" of A
			6	SA ES2	0 +r → "u" of A
			7	SA ES3	0 EL+r → "u" of A
			8	MJ 0	ES
			9	TP FC36	A Equality not met indicator → A (i.e. A ₇₁ = 1)
			10	MJ 0	ES
			11	RA RA5	FC3 Adv. dummy tally for Exp. List by 1 in "u" and "v"
			12	TJ LV5	ES Expanded List too long? no; to exit
			13	RJ WA	WA1 Yes; type sentence number
			14	TP TO	UP3 Codeword → alarm print
			15	RJ UP2	UP Alarm-Exp. List too long [type: SENTENCE TOO LONG.]
			16	MJ 0	BQ6 Rewind tapes and stop
	17	CA ES20			

				Dimension List Search
(189)	0	IA DS	[30000]	Exit
	1	MJ 0	EP1	Search Dimension List (preset from f ₆)
	2	RP [30000]	DS3	Callword in Dimension List? Alarm #1 if no.
	3	EJ DL	DS3	no.
	4	SN Q	17	Yes; -jn+r → "u" of A
	5	SA DS1	0	+r → "u" of A
	6	SA DS2	0	DL+r → "u" of A
	7	MJ 0	DS	
	8	CA DS7		

Subroutine to Decrease and Check
Partial Result Counter

(190)	0	IA PR RS RA6	FC1	Decrease partial result counter to new P.R. symbol
	1	TJ LV6	PR3	Has P.R. counter reached minimum value? "v" if yes
	2	MJ 0	[30000]	No; to exit
	3	RJ WA	WA1	Yes; type sentence number
	4	TP TO	UP3	Codeword → alarm print
	5	RJ UP2	UP	Alarm-P.R. counter below minimum [type: SENTENCE TOO LONG]
	6	MJ 0 CA PR7	BQ6	

Subroutine to Check for 6l___ Type
Operands in Dummy Instruction

(191)	0	IA EK TP FC77	Q	Mask → Q
	1	QT WT5	A	First two octal digits of "u" and "v" operands to "A"
	2	EJ FC100	EK23	"u" and "v" operands = 6l___type? yes; take "v"
	3	TP FC104	Q	Mask for first two octal digits of "v" → Q
	4	QT WT5	A	First 2 octal digits of "v" operand to "A"
(192)	5	EJ FC102	EK20	"v" operand = 6l___type? yes; take "v"
	6	TP FC103	Q	Mask for first two octal digits of "u" to "Q"
	7	QT WT5	A	
	10	EJ FC101	EK12	"u" operand = 6l___type? yes; take "v"
	11	MJ 0	EK25	To exit
(193)	12	RA CT10	FC115	Adv. cond ind. by 33 in op. code
	13	MJ 0	EK21	
(194)	14	TP FC104	Q	Mask for first two octal digits of "v" to "Q"
	15	QT WT5	A	First 2 octal digits of "v" operand to "A"
	16	EJ FC102	EK12	"v" operand = 6l___type? yes; take "v"
	17	MJ 0	EK25	To exit
	20	RA CT10	FC105	Adv. cond. ind. by 35 in op. code
	21	RA RA7	FC3	Adv. Nrp by 1 in "u" and "v"
	22	MJ 0	EK25	To exit
(195)	23	RA RA7	FC4	Adv. Nrp by 2 in "u" and "v"
	24	RA CT10	FC106	Adv. cond. ind. by 31 in op. code
	25	MJ 0 CA EK26	[30000]	Exit

				Subroutine
		IA	PP	to Enter Current Partial Result Symbol
(196)	0	RJ	ES ES12	in Expanded List and Operand List
	1	TP	RA5 RA4	Advance dummy tally (D) by 1 in "u" and "v"
	2	RJ	PR2 PR	Set $\gamma = D$ (advance γ to add Dummy List to Exp. List)
	3	TV	RA PP4	Decrease P.R. counter \rightarrow new partial result (P.R. in A)
	4	TP	A [30000]	Available address in Operand List (β) \rightarrow "v" of NI
(197)	5	TV	RA4 PP6	P.R. value \rightarrow Operand List
	6	TP	A [30000]	Next address in Exp. List \rightarrow "v" of NI
	7	RJ	BR BR4	P.R. value \rightarrow Expanded List
	10	MJ	0 [30000]	Adv. add. in Operand List (β) by 1 in "u" and "v"
		CA	PP11	

				Subroutine to Store
		IA	SR	Partial Result Symbol for Subscript Operation in Expanded List and Operand List
(198)	0	TV	[30000] WT1	Last subscript \rightarrow "v" of WT1
	1	TP	WT1 A	Last subscript \rightarrow "v" of A
	2	EJ	RA6 SR7	P.R. counter = last subscript (i.e. subscript in A)
	3	RA	RA7 FC3	No, advance Nrp by one
(198A)	4	RA	RA7 CT7	Advance Nrp by increment (I)
	5	MJ	0 SR11	
	6	0	[30000] [30000]	Vacant
(199)	7	RA	CT10 FC5	Set condition indicator \rightarrow one (s.s. in A)
(199A)	10	RJ	LA LA6	Enter P.R. value in "A" list
(200)	11	RJ	ES ES12	Advance dummy tally (D) by 1 in "u" and "v" to count P.R.
	12	TP	RA5 RA4	Set $\gamma = D$ (advance γ to add Dummy List to Exp. List)
(201)	13	RJ	PR2 PR	Decrease P.R. counter \rightarrow new partial result (P.R. in A)
	14	TV	RA SR15	Available address in Operand List (β) to "v" of NI
	15	TP	A [30000]	P.R. value \rightarrow Operand List
	16	TV	RA4 SR17	Next address in Expanded List \rightarrow "v" of NI
	17	TP	A [30000]	P.R. value \rightarrow Expanded List
(202)	20	TV	WT4 SR23	Dummy inst. address \rightarrow "v" of QT
	21	RA	SR23 FC1	Advance add. of QS \rightarrow word following dummy inst.
	22	TP	FC36 Q	Mask for op. code \rightarrow Q
	23	QS	CT10 [30000]	Condition indicator \rightarrow op. code of word following dummy inst.
	24	RJ	BR BR4	Advance address in Operand List (β) by 1 in "u" and "v"
	25	MJ	0 [30000]	
		CA	SR26	

	IA	FC		Fixed Constants
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	0	2	2	
5	1	0	0	Subscript for "u" in A
6	2	0	0	Subscript for "v" in A
7	3	0	0	Operand for "u" in Q
10	4	0	0	Operand for "v" in Q
11	5	0	0	
12	10	0	0	
13	12	0	0	
14	17	0	0	
15	20	0	0	
16	22	0	0	
17	25	0	0	
20	30	0	0	
21	40	0	0	
22	70	0	0	
23	0	0	2	
24	0	3	3	
25	0	4	4	
26	0	5	5	
27	0	6	6	
30	0	0	7	
31	0	0	10	
32	0	0	77777	
33	0	77777	0	
34	0	07777	0	
35	0	77777	77777	
36	77	0	0	
37	77	0	77777	
	IA	FC40		
40	77	77777	0	
41	07	77777	77777	
42	07	77000	0	
43	0	0	61000	
44	0	0	62000	
45	0	0	64000	
46	0	0	70000	
47	0	0	73000	
50	0	0	73777	
51	0	0	74000	
52	0	0	76777	
53	0	0	77000	
54	0	74777	0	
55	0	74000	0	
56	0	76777	0	

57	0	0	3	
60	0	0	76000	
61	0	0	75000	
62	11	0	0	
63	0	0	7	
64	0	0	50012	General power library routine callword
65	20	14000	0	Floating point-one
66	0	0	50051	Square root library routine callword
67	0	0	50022	Int. x^y where $ y > 29$
70	6	0	0	
71	0	0	31000	
72	0	31000	0	
73	0	0	700	
74	0	0	77	
75	37	40000	0	
76	0	0	74777	
77	0	77000	77000	
	IA	FC100		
100	0	61000	61000	
101	0	61000	0	
102	0	0	61000	
103	0	77000	0	
104	0	0	77000	
105	35	0	0	
106	31	0	0	
107	0	01000	01000	
110	0	20000	20000	
111	0	0	4	
112	0	0	50000	
113	MJ	0	0	
114	77	077777	777777	
115	33	0	0	
116	0	777	777	
117	0	0	16000	
120	0	0	16100	
121	0	7	7	
122	77	777777	777776	
	CA	FC123		

		Op. Codes Contain Operation			
IA	RC	Symbols for Expanded List;		Relative	Constants in "v"
0	0	0	PN34	to set	(H) → (H2)
1	37	0	PN1	Subscript operator	to set (H) → (H1)
2	52	0	PN15	Floating subtract	to set (G) → (G1)
3	54	0	PN17	Floating divide	to set (G) → (G2)
4	61	0	PN45	Library operator	to set (J) → (J1)
5	62	0	PN47	Floating unary minus	to set (J) → (J2)
6	64	0	PN51	Floating absolute value	to set (H) → (H3)
7	66	0	SS	POW +2	to set (H) → (I)
10	67	0	PN62	POW -2	to set (M) → (M1)
11	70	0	PN64	POW +3	to set (M) → (M2)
12	71	0	LN2	POW -3	to set (S) → (S2)
13	72	0	LN	POW ½	to set (S) → (S1)
14	73	0	LN4	POW -½	to set (S) → (S3)
15	74	0	LN12	POW (4 to 63)	to set (N) → (N2)
16	75	0	RR50	POW (-4 to -63)	to set (T) → (T1)
17	76	0	IQ31	POW -1	to set (T) → (T2)
20	77	0	PN74	Storage operator	to set (M) → (M3)
21	41	0	0	Floating plus	
22	53	0	0	Floating multiply	
23	55	0	0	Fixed plus	
24	56	0	0	Fixed subtract	
25	57	0	0	Fixed multiply	
26	60	0	0	Fixed divide	
27	63	0	0	Fixed unary minus	
30	65	0	0	Fixed absolute value	
31	0	0	S044		
32	0	0	S054		
CA	RC33				

	IA	TO		Alarm Text
0	40	T01	3	
1	65	30506	63050	S E N T E N
2	26	30016	65151	C E Δ T O O
3	01	46515	03222	Δ L O N G .
	CA	T04		

		IA	IA	Initial Addresses	
To preset RA	0	0	BL	BL	β initial (initial add. Operand List)
To preset RA1	1	0	20000	FL2	Op. File 1 item (jn to search in "u" - init. cross ref. item in "v")
To preset RA2	2	0	20000	XQ	"Q" list (jn to search in "u" - init. add. "Q" List in "v")
To preset RA3	3	0	20000	XA	"A" list (jn to search in "u" - init. add. "A" List in "v")
To preset RA4	4	0	20000	EL1	Exp. list (jn to search in "u" - init. avail. add. -1 in "v")
To preset RA6	5	0	0	31000	Initial P.R. value in "v" +1
To preset RA7	6	0	01001	01001	Init. #lines in running prog +1000 in "u" and "v" (counts MJ exit)
To preset RA8	7	0	20000	RL	Red. P.R. List (jn to search in "u" - init. add. in Red. P.R. List in "v")
To preset SS3	10	0	SL3	0	α initial (initial address in Sorted List)
	11	0	EL2	0	Init. add. in Exp. List +2
		CA	IA12		

		IA	LV	Limiting Addresses for Lists, etc.	
	0	0	BL177	BL177	Max. address in Oper. List in "u" and "v" (max. β)
	1	0	20077	RL77	Max. jn in "u" and max. address in "v" for redundant P.R. List
	2	0	20175	FL177	Max. jn in "u" and max. address in "v" for Op. File 1 item
	3	0	20177	XQ177	Max jn in "u" and max. address in "v" for "Q" List
	4	0	20177	XA177	Max. jn in "u" and max. address in "v" for "A" List
	5	0	20675	EL677	Max. jn in "u" and max. address in "v" for Expanded List
	6	0	0	30000	Minimum P.R. value in "v"
	7	0	02002	02002	Max. #lines object prog. body (incl. jump to exit) +1001
	10	0	0	62000	Dummy callword for function input region
	11	0	0	63000	Dummy callword for pseudo operation input region
		CA	LV12		

Subroutine to Store in List 1, Callword of Library Routine and if Fixed Library Routine, Callwords of its Cross-references

	IA	LS		
203	0	MJ	0	[30000]
	1	TU	LS26	LS13
	2	TP	FC	CT13
	3	EJ	LS25	LS11
	4	EJ	LS21	LS12
	5	RP	20003	LS14
	6	EJ	LS22	LS7
	7	RJ	LR	LR1
	10	MJ	0	LS13
204	11	RA	CT13	FC23
205	12	RA	CT13	FC57
206	13	TP	[30000]	A
207	14	RJ	LR	LR1
	15	RA	LS13	FC2
	16	IJ	CT13	LS13
	17	MJ	0	LS
	20	0	50002	0
	21	0	50022	0
	22	0	50031	0
	23	0	50041	0
	24	0	50051	0
	25	0	50012	0
	26	0	LS20	0
		CA	LS27	

Zero → index C₃
 Callword = 50012? yes; to LS11
 Callword = 50022? yes; to LS12
 Callword = 50031, 50041, or 50051?
 Yes; to LS7
 50031, 50041, or 50051 callword to List 1
 50002 callword to list 1
 Advance index by 2 in "v"
 Advance index by 3 in "v"
 Callword → A
 Callword to list 1
 Advance by 1 in "u" to address of next callword
 More callwords to store in list 1? yes; to LS13
 No; to exit

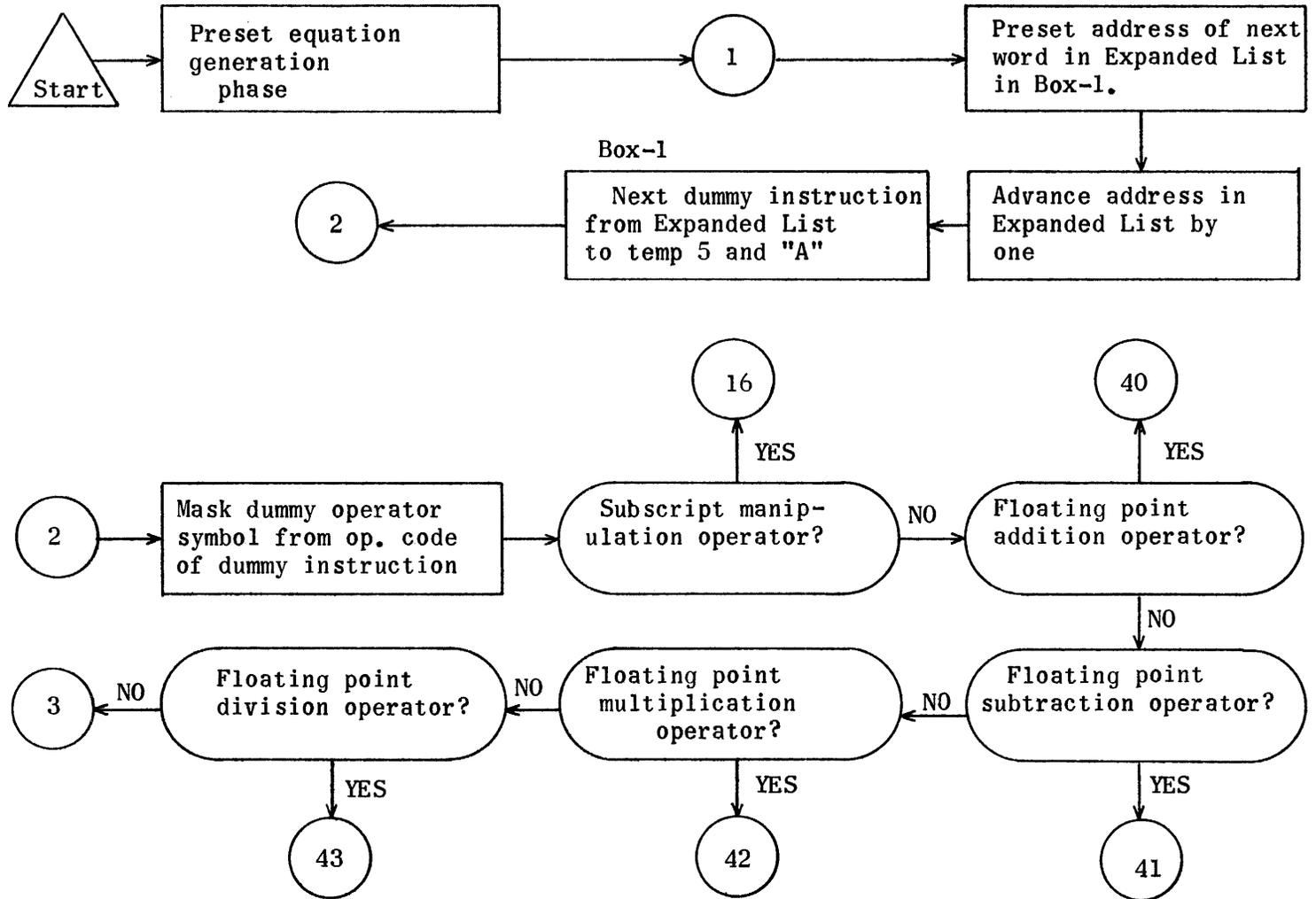
Explanation of working Temporaries (WT)

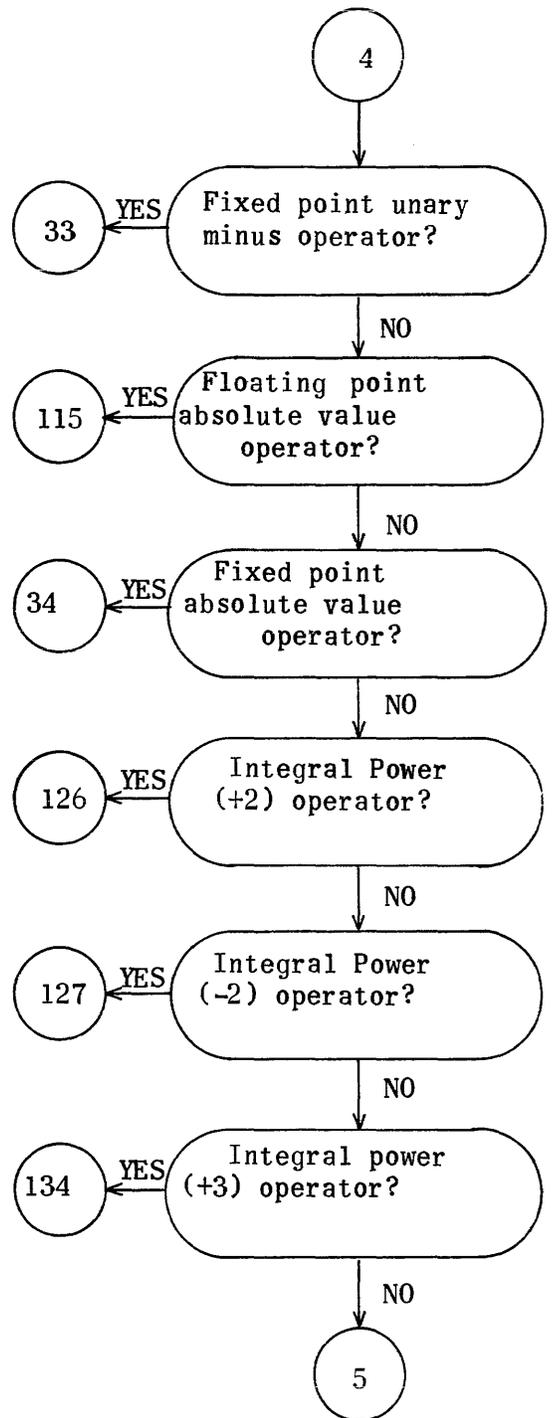
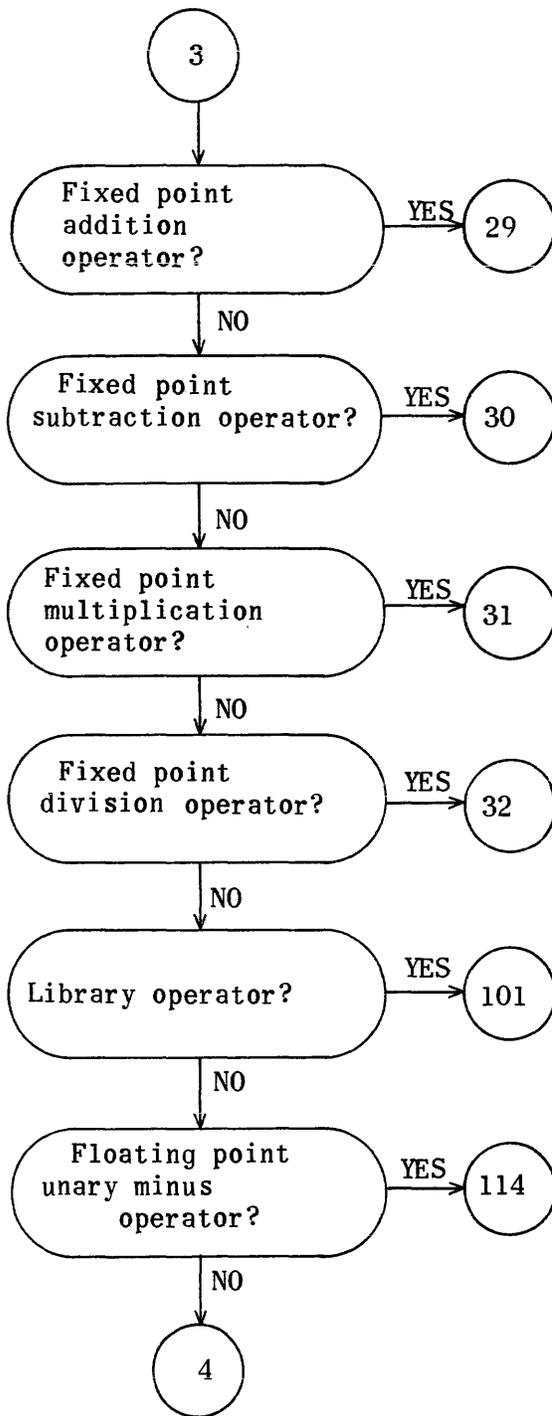
WT0	0	[30000]	0	Temp 0 - op. code and "v" always Zero
1	0	0	[30000]	Temp 1 - op. code and "u" always Zero
2	[—	—	—]	Temp 2
3	[—	—	—]	Symbol temp Floating point inst.
4	[—	—	—]	Dummy inst. address Register indicator
5	[—	—	—]	Dummy instruction
6	[—	—	—]	Subscript word following dummy instruction

Explanation of Counters (CT)				
CT0	0	[—]	[—]	Crc-no. relative constants
1	0	[—]	[—]	Crpt-no. ten lines in running prog. body
2	0	[—]	[—]	Crct-no. ten lines in relative constant region
3	0	[—]	[—]	Trp-subtally of no. lines in running prog.
4	0	[—]	[—]	Trc-subtally of no. relative constants
5	0	[—]	[—]	Trpt-subtally of no. ten lines in running prog. body
6	0	[—]	[—]	Trct-subtally of no. ten lines in rel. const. region
7	0	[—]	[—]	Increment (I)
10	[—]	—	—	Condition indicator
11	0	0	[—]	Index counter (C ₁)
12	0	0	[—]	Index counter (C ₂)
13	0	0	[—]	Index counter (C ₃)

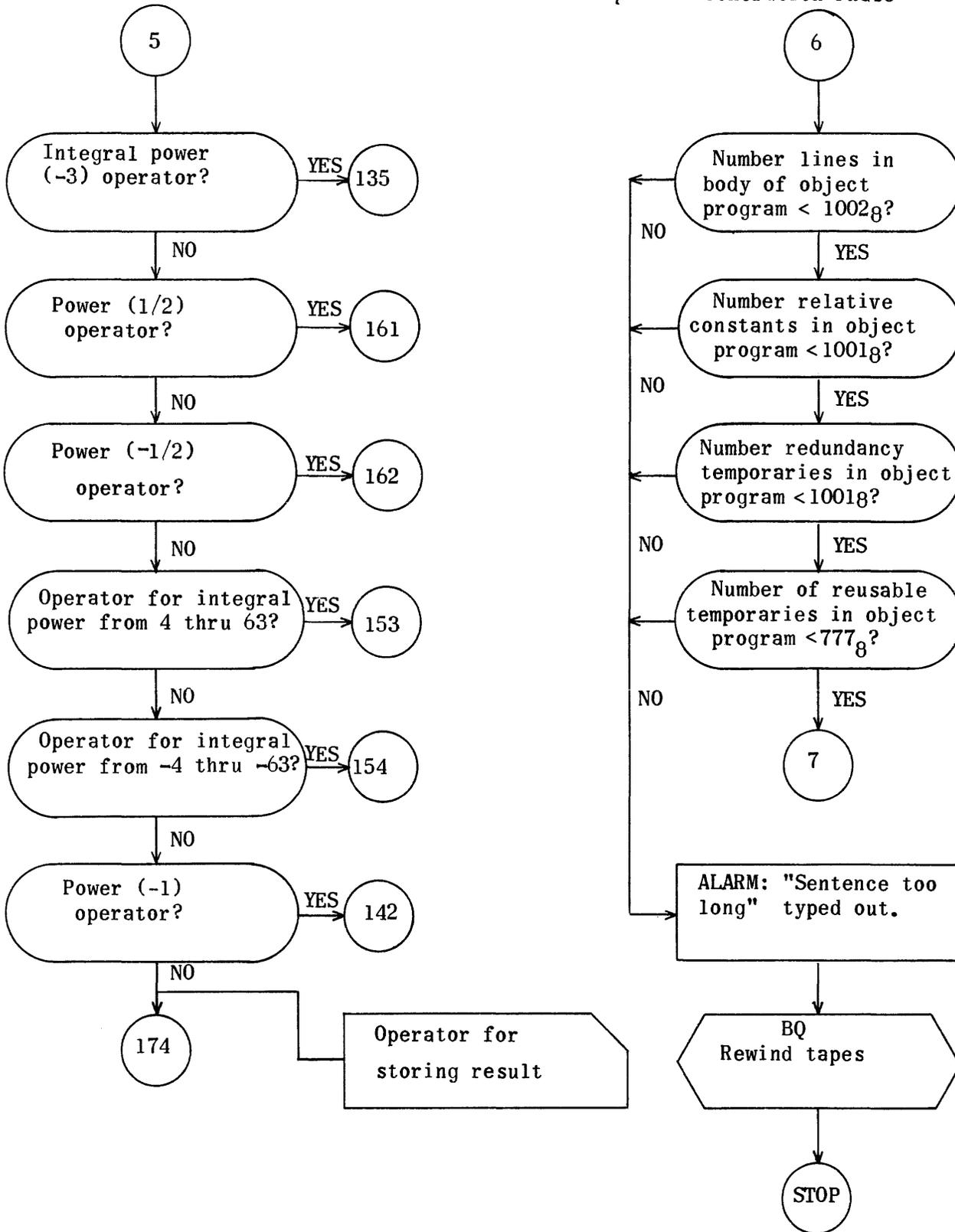
List of running (current) Addresses in Lists (RA)				
RA0	[—]	—	—	β (available open address in Operand List in "u" and "v")
1	[—]	—	—	Op. File 1 tally (jn in "u"-available address in "v")
2	[—]	—	—	"Q" List tally (jn in "u"-available address in "v")
3	[—]	—	—	"A" List tally (jn in "u"-available address in "v")
4	[—]	—	—	γ -Expanded List tally (jn in "u"-last used address in "v")
5	[—]	—	—	Dummy tally (D) for Expanded List (same format as γ)
6	[—]	—	—	Partial result (P.R.) counter (current P.R. in "v")
7	[—]	—	—	Tally of number of lines in running program +1000
10	[—]	—	—	Redundant P.R. List tally (jn in "u"-available address in "v")

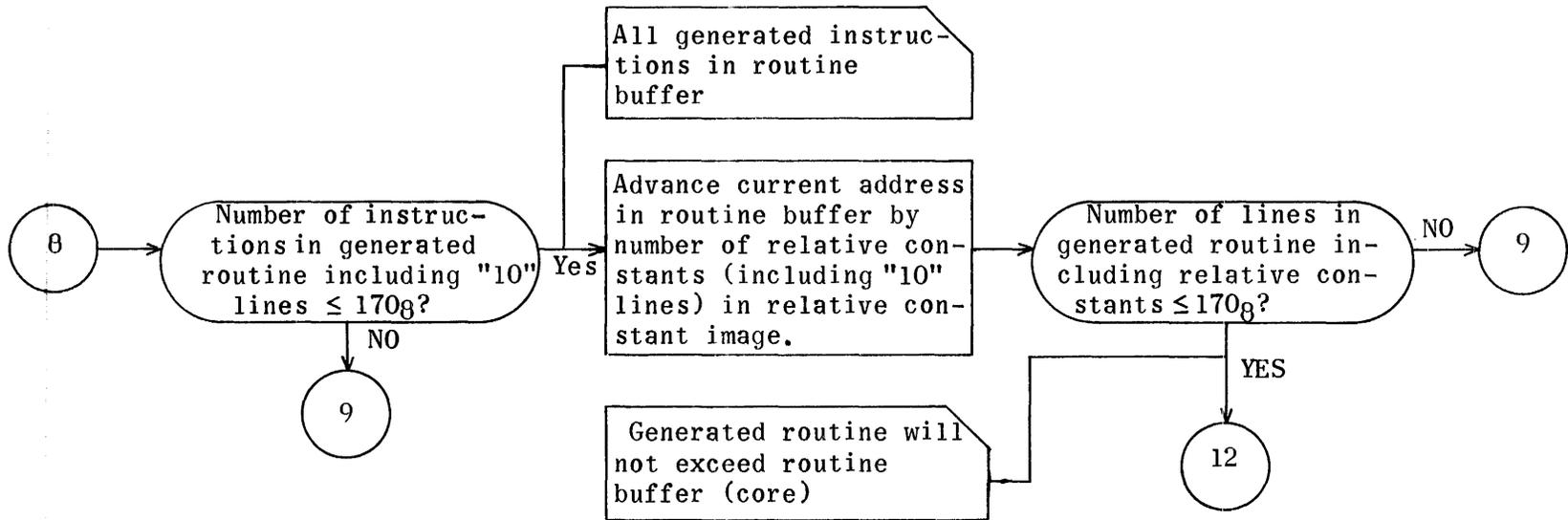
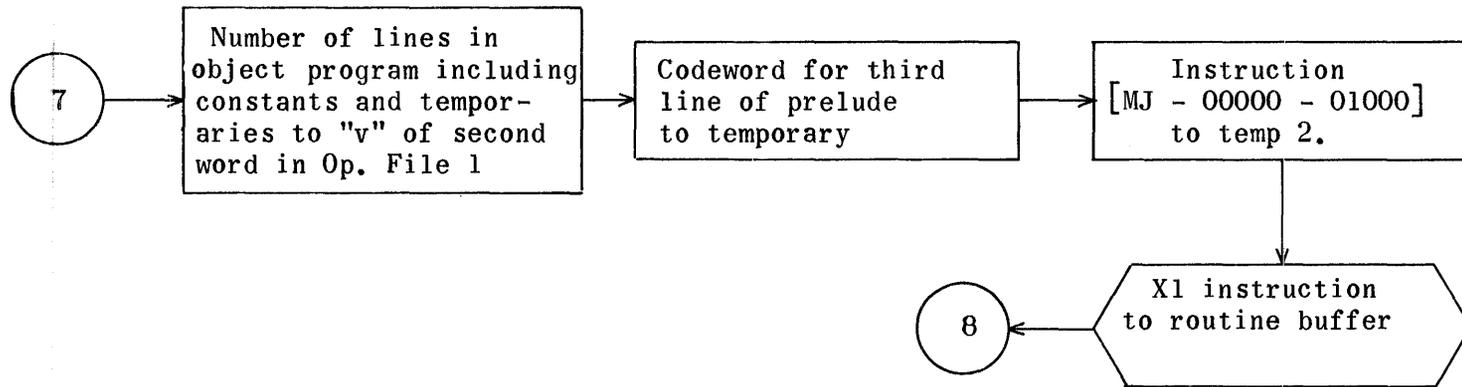
Equation Generation No. 3 Flow Charts
Equation Generation Phase

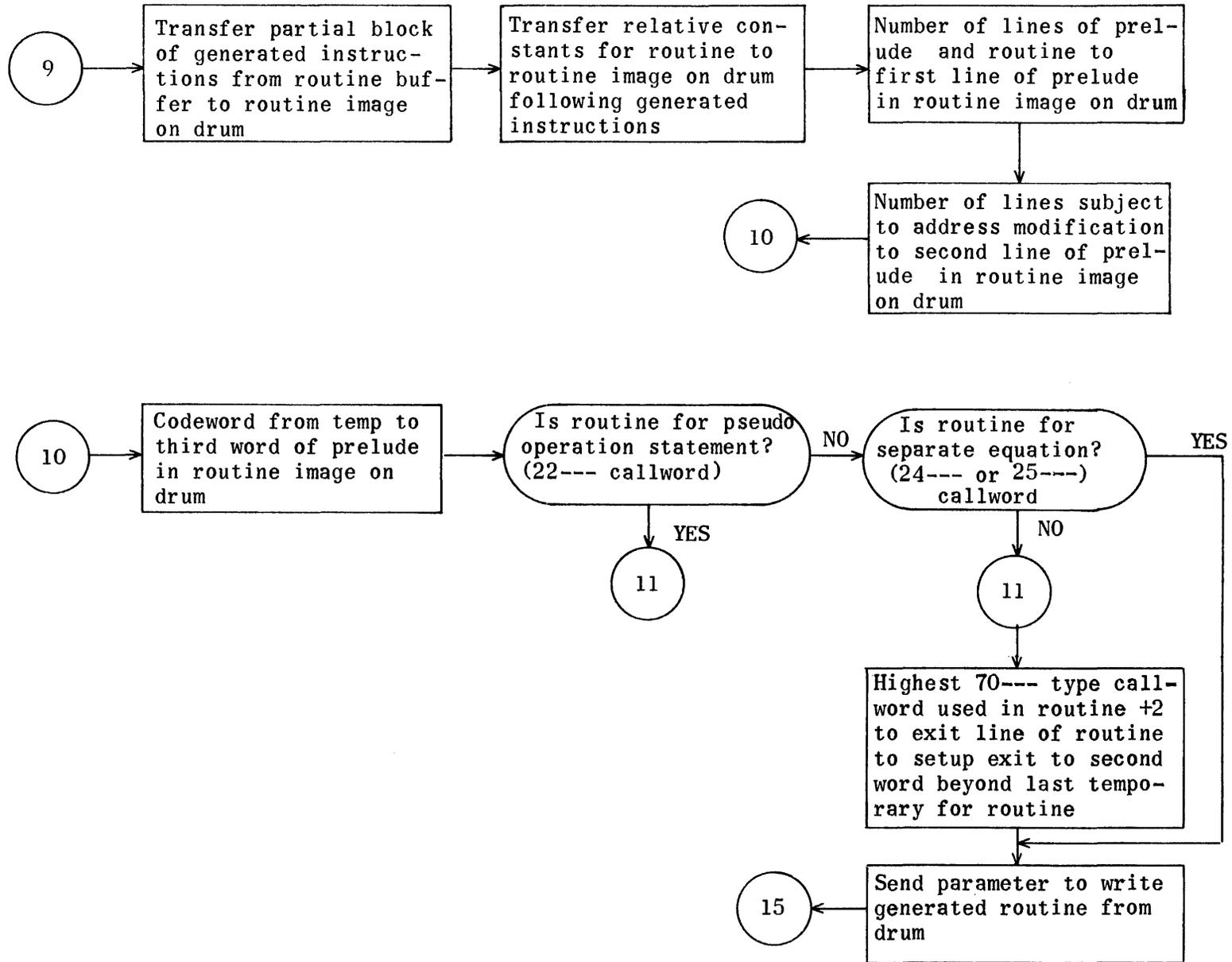


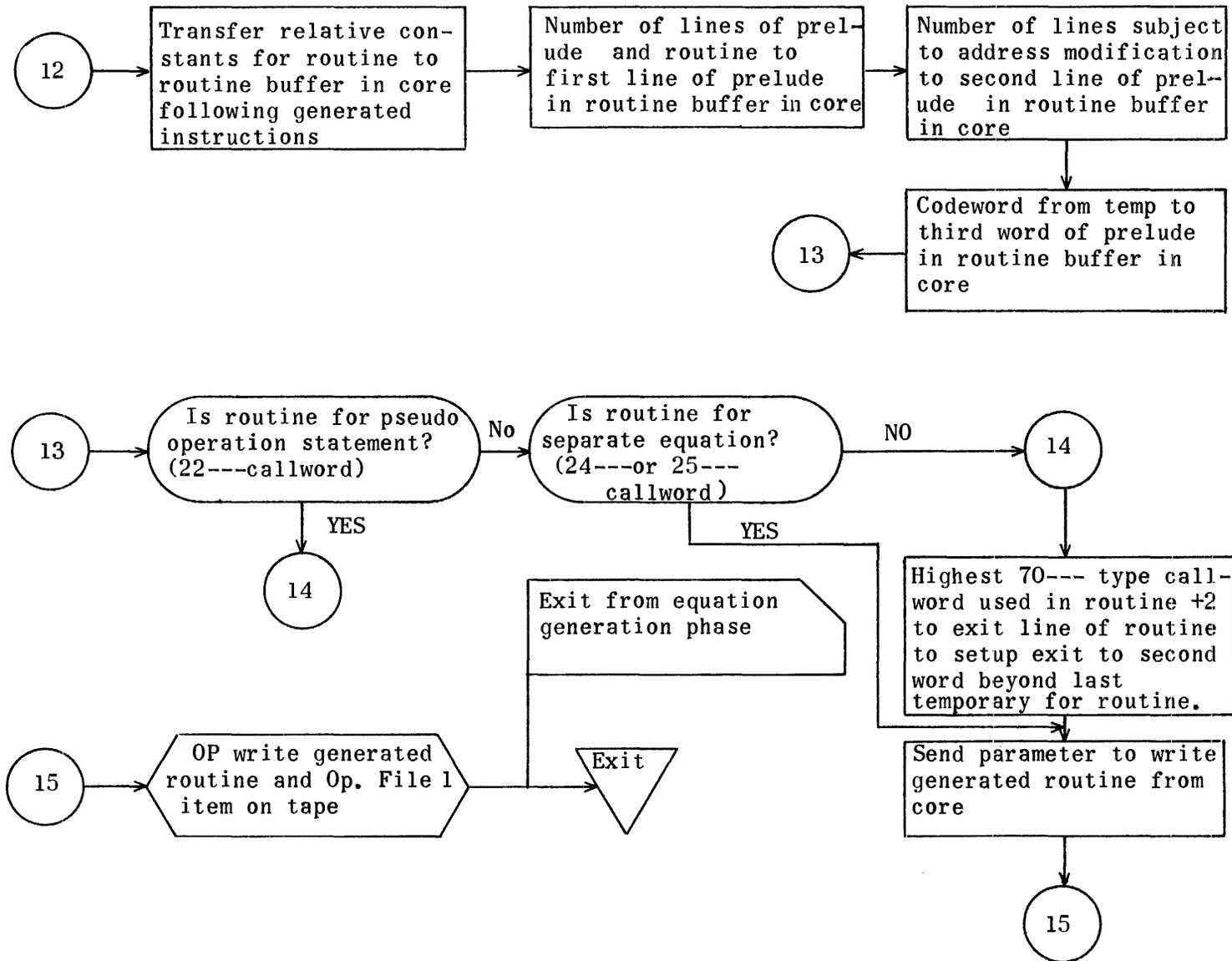


End Equation Generation Phase

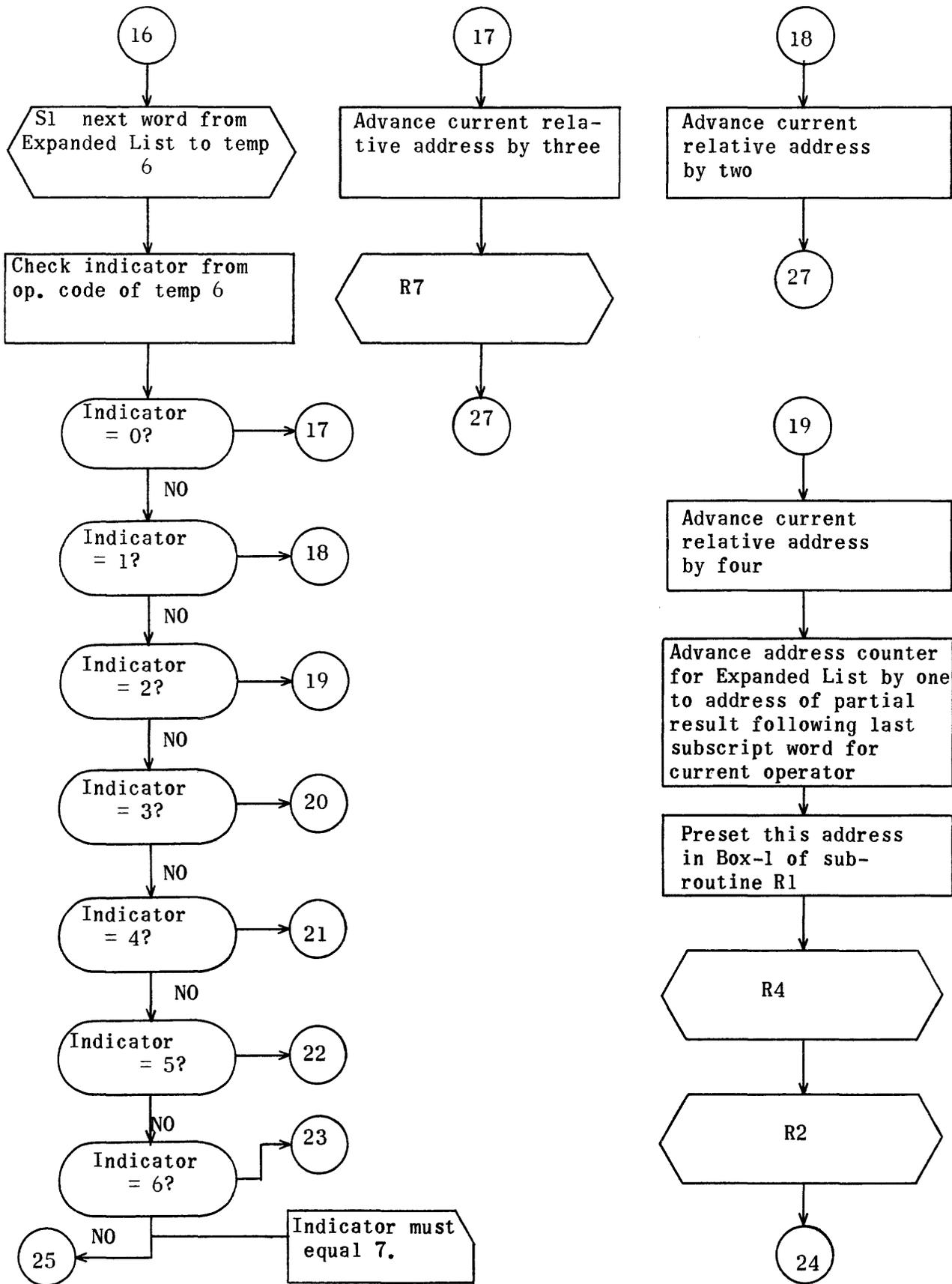


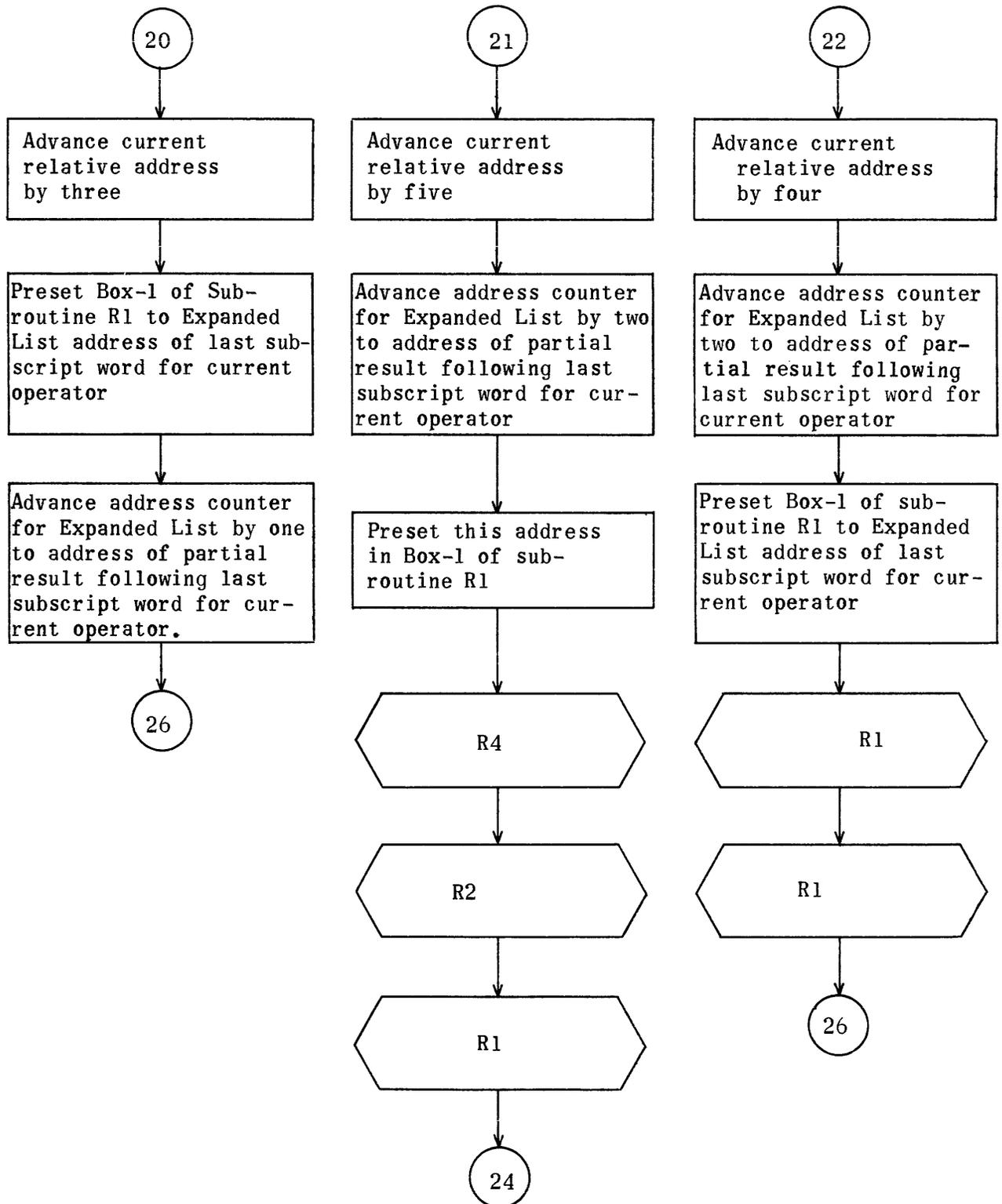


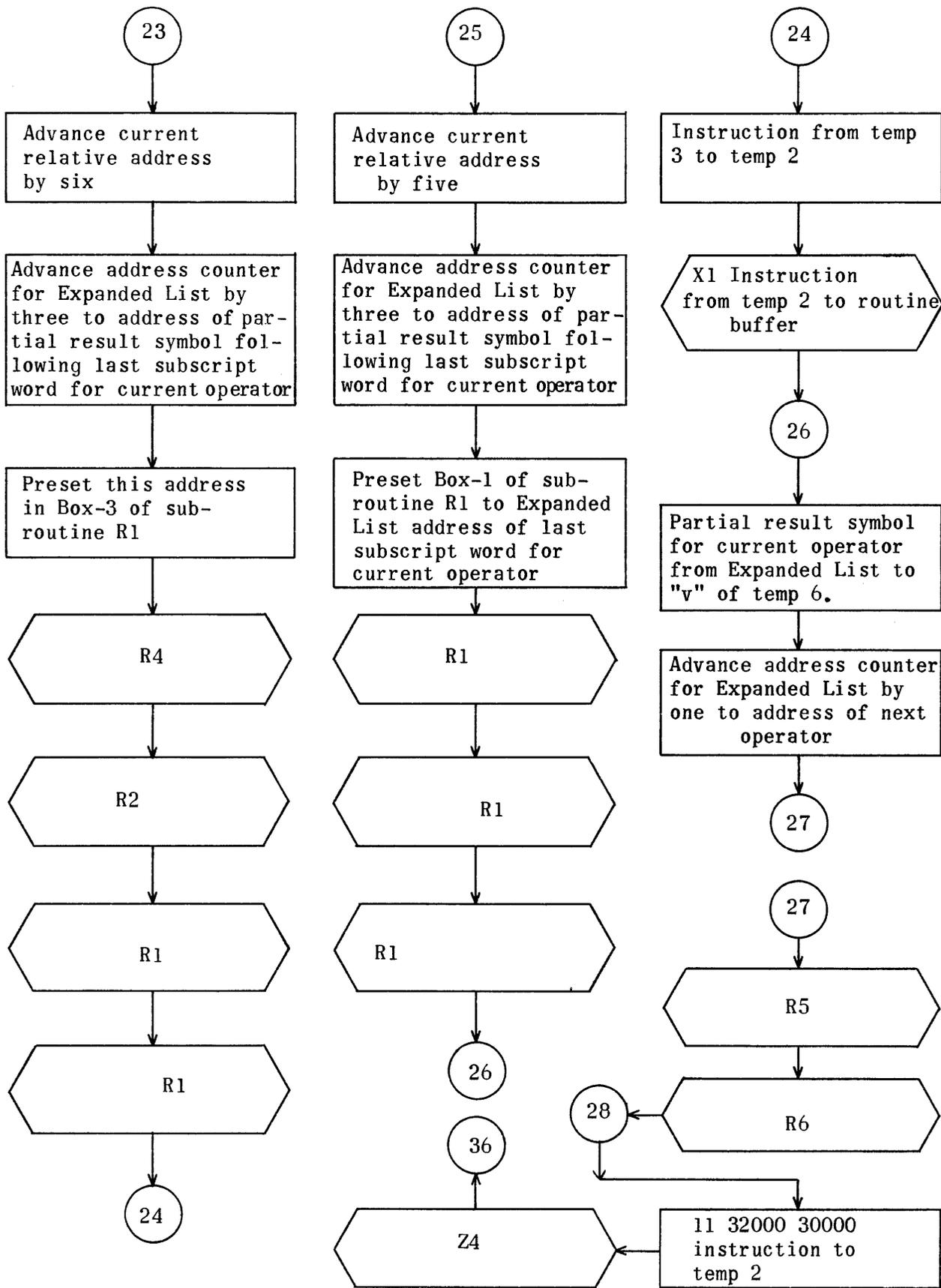




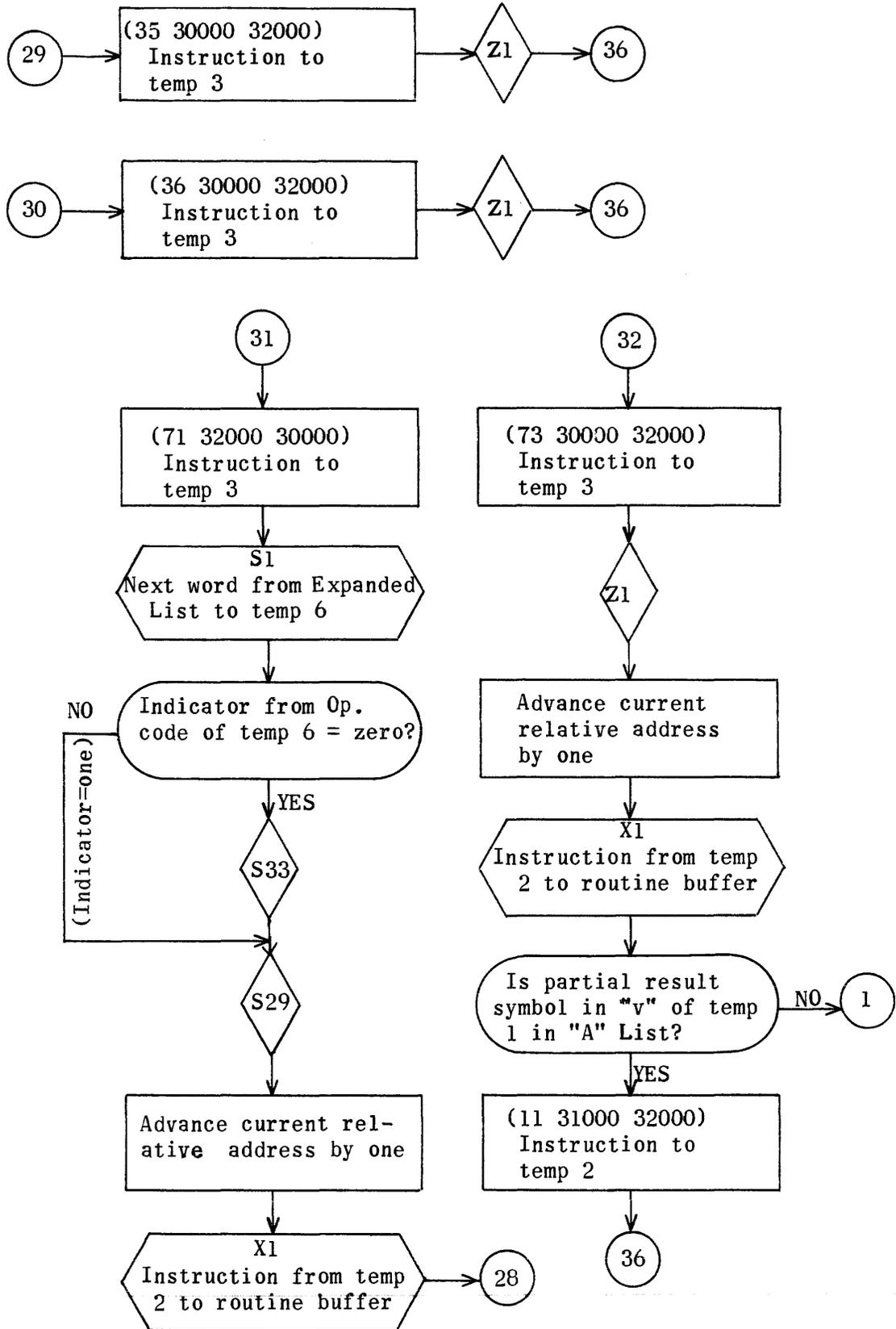
Equation Generation Phase (Subscript Manipulation Operator)



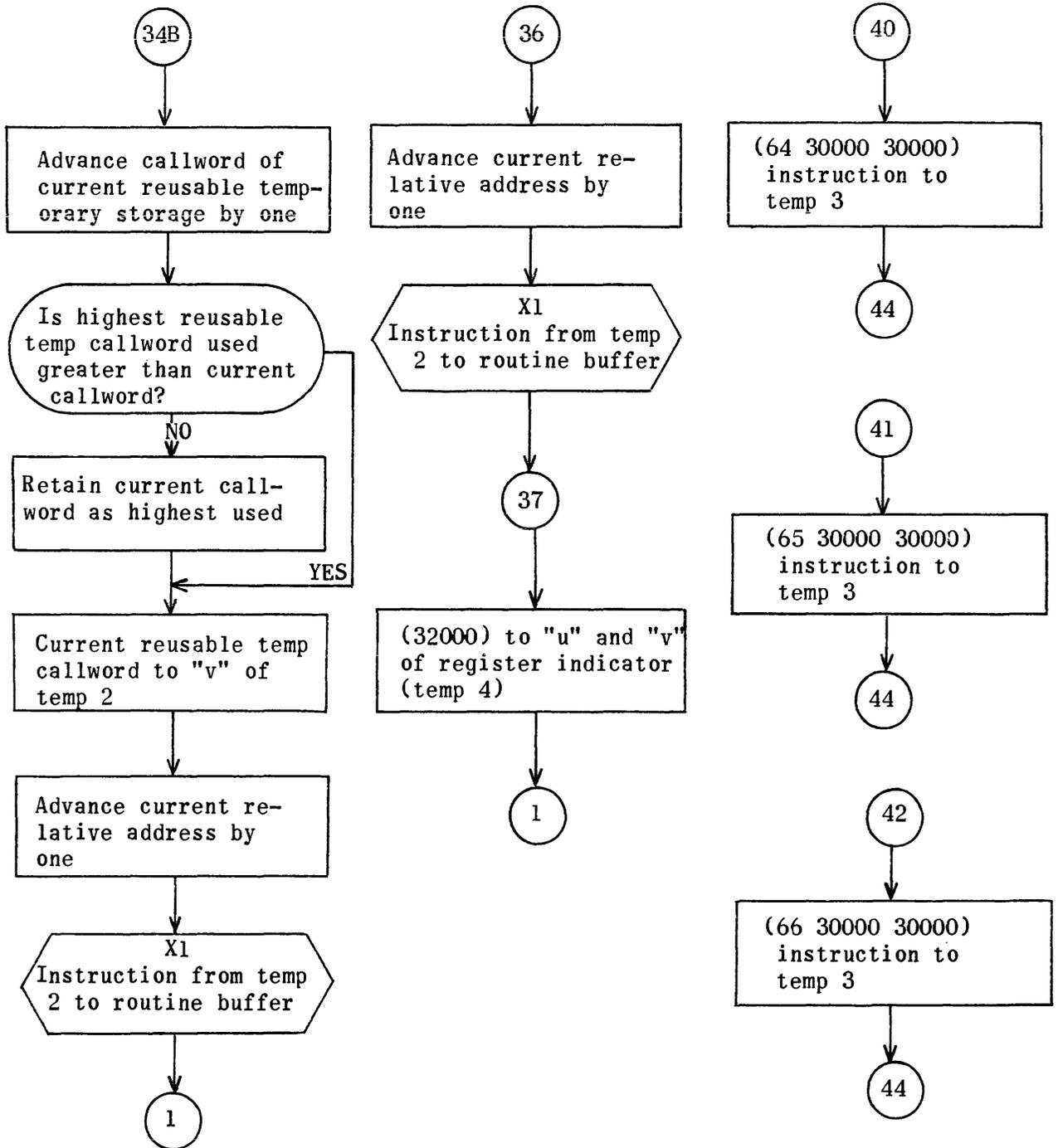


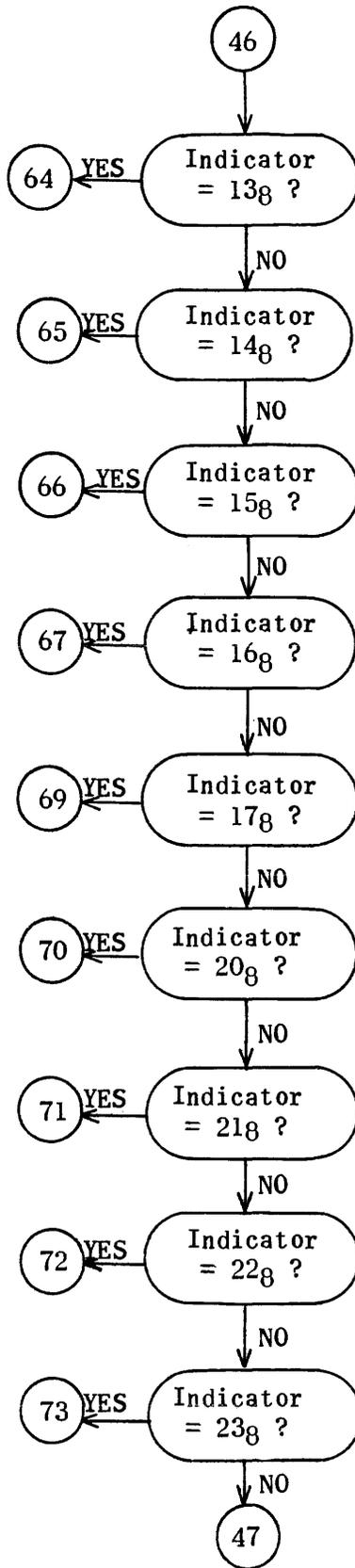
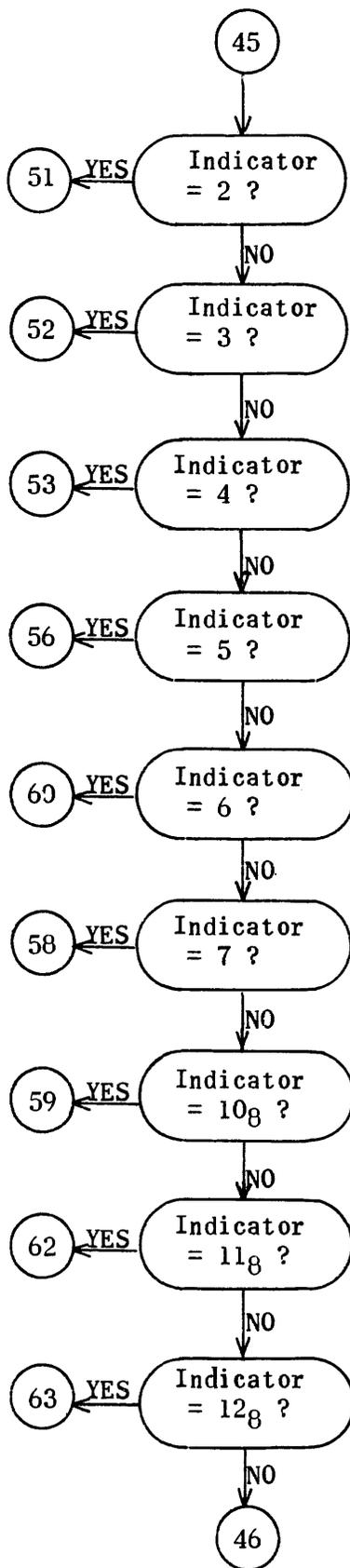
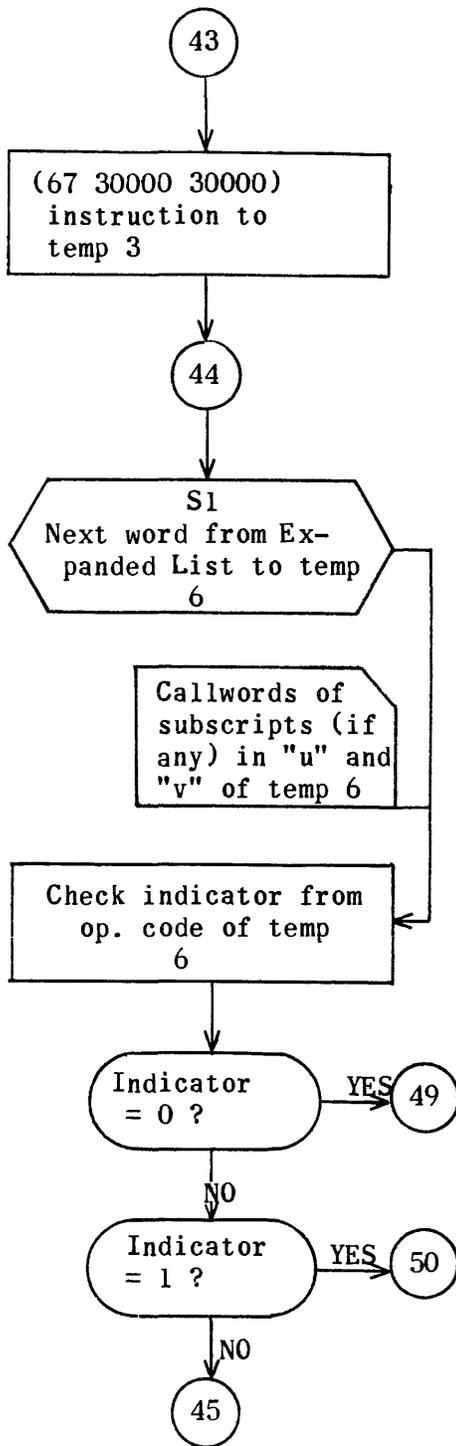


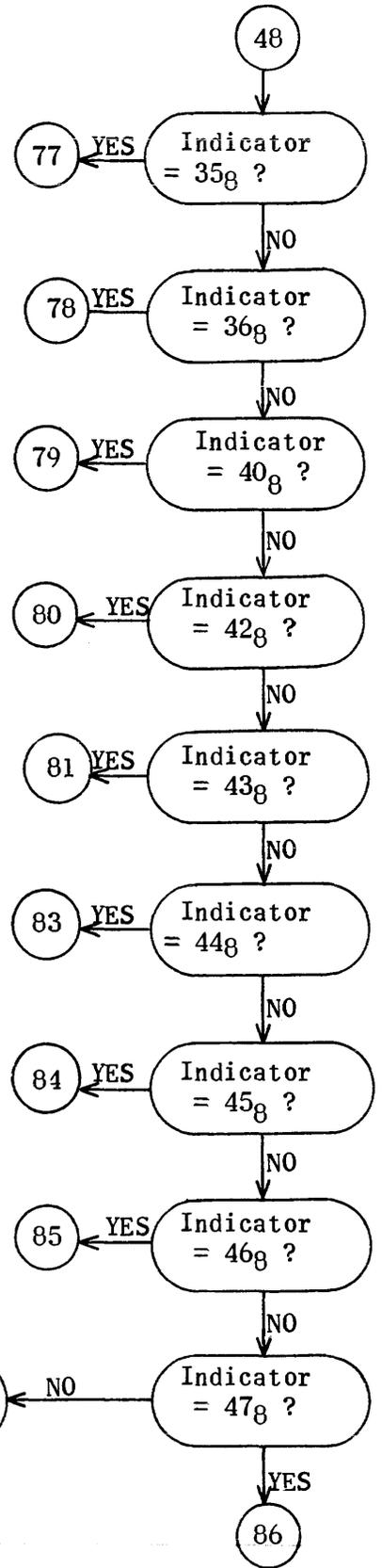
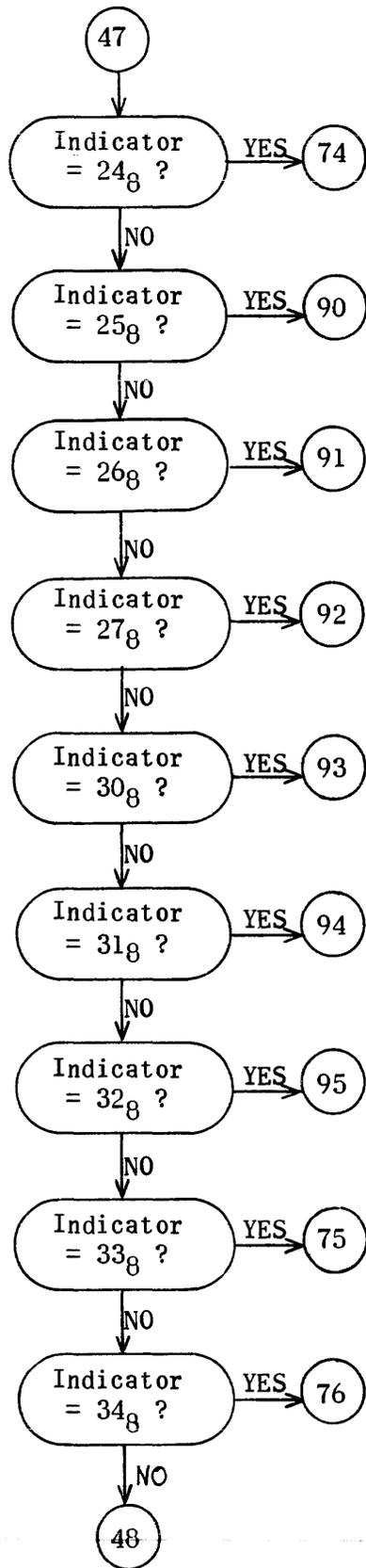
Equation Generation Phase (Fixed Point Operators)

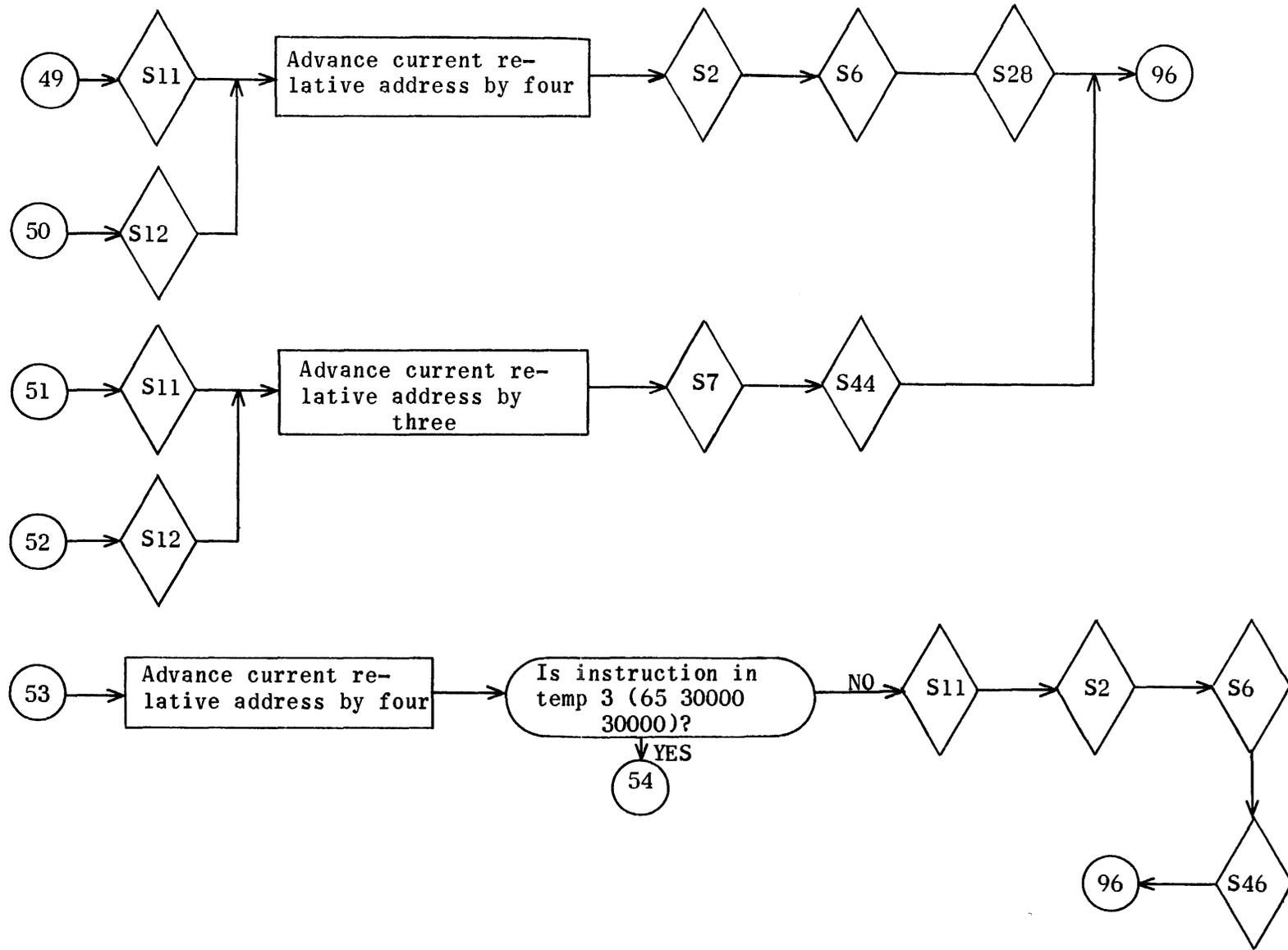


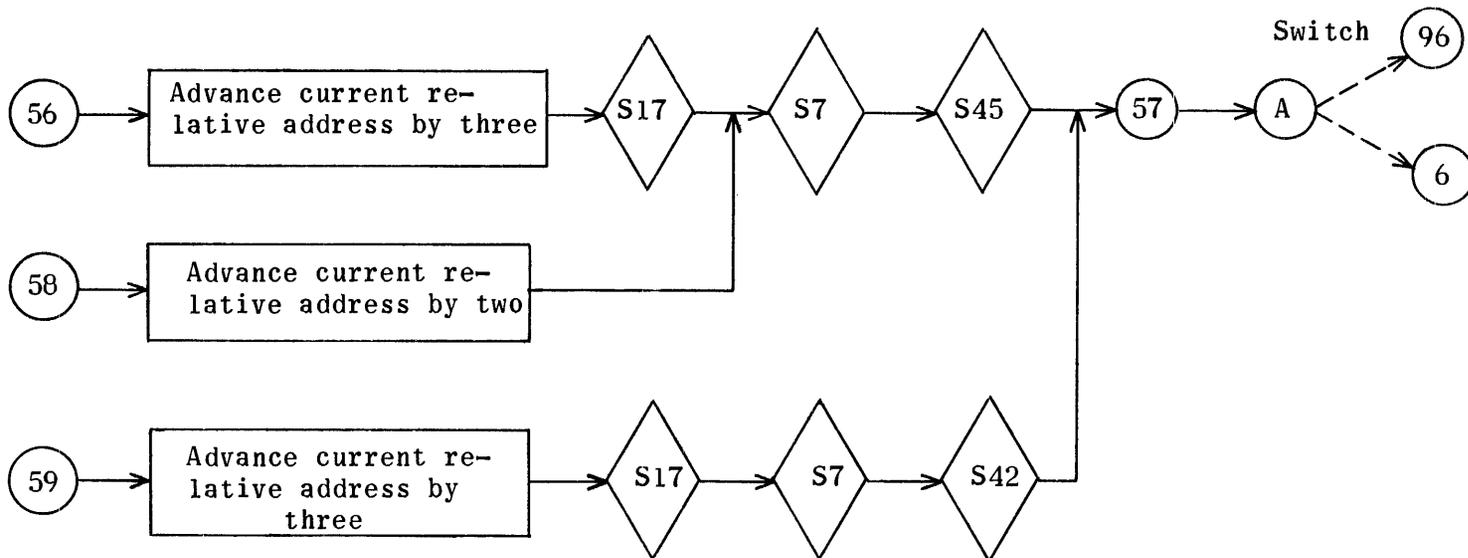
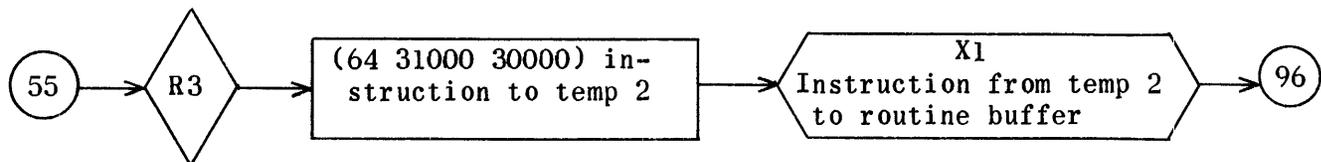
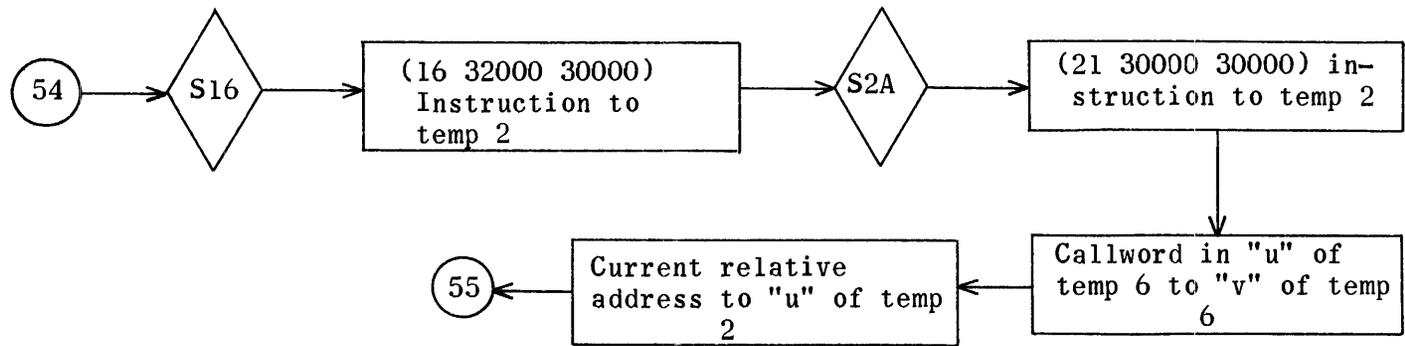
Equation Generation Phase (Fixed Point Operators
and Floating Point Binary Operators)

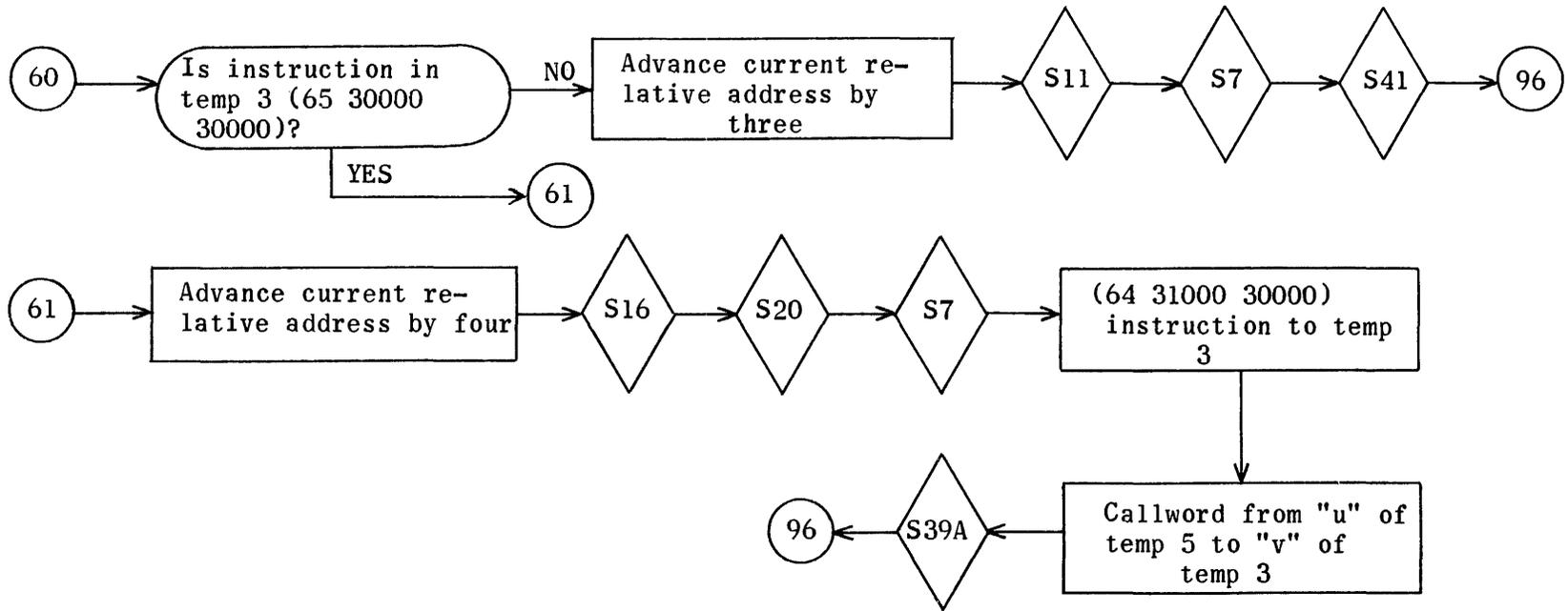


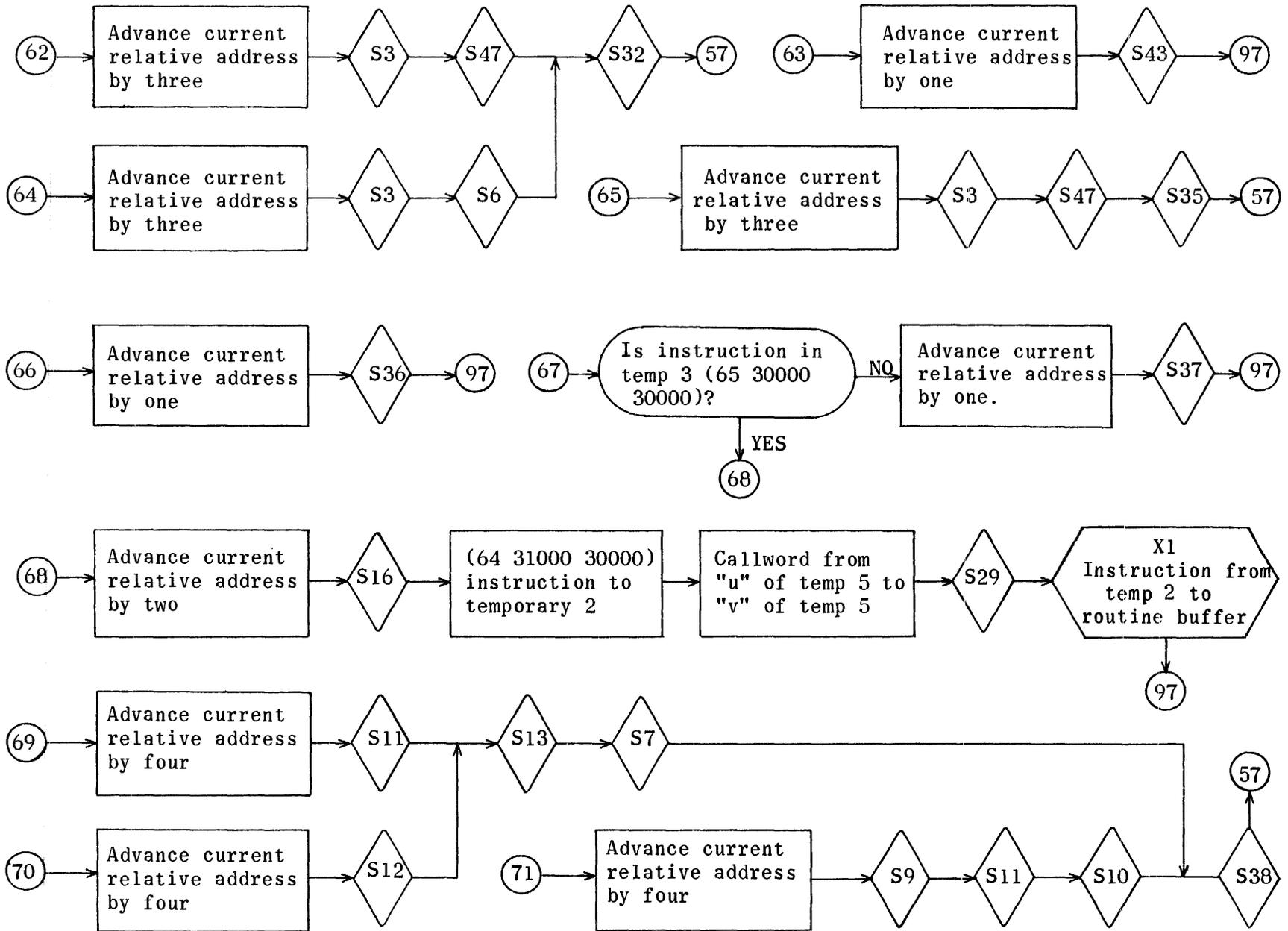


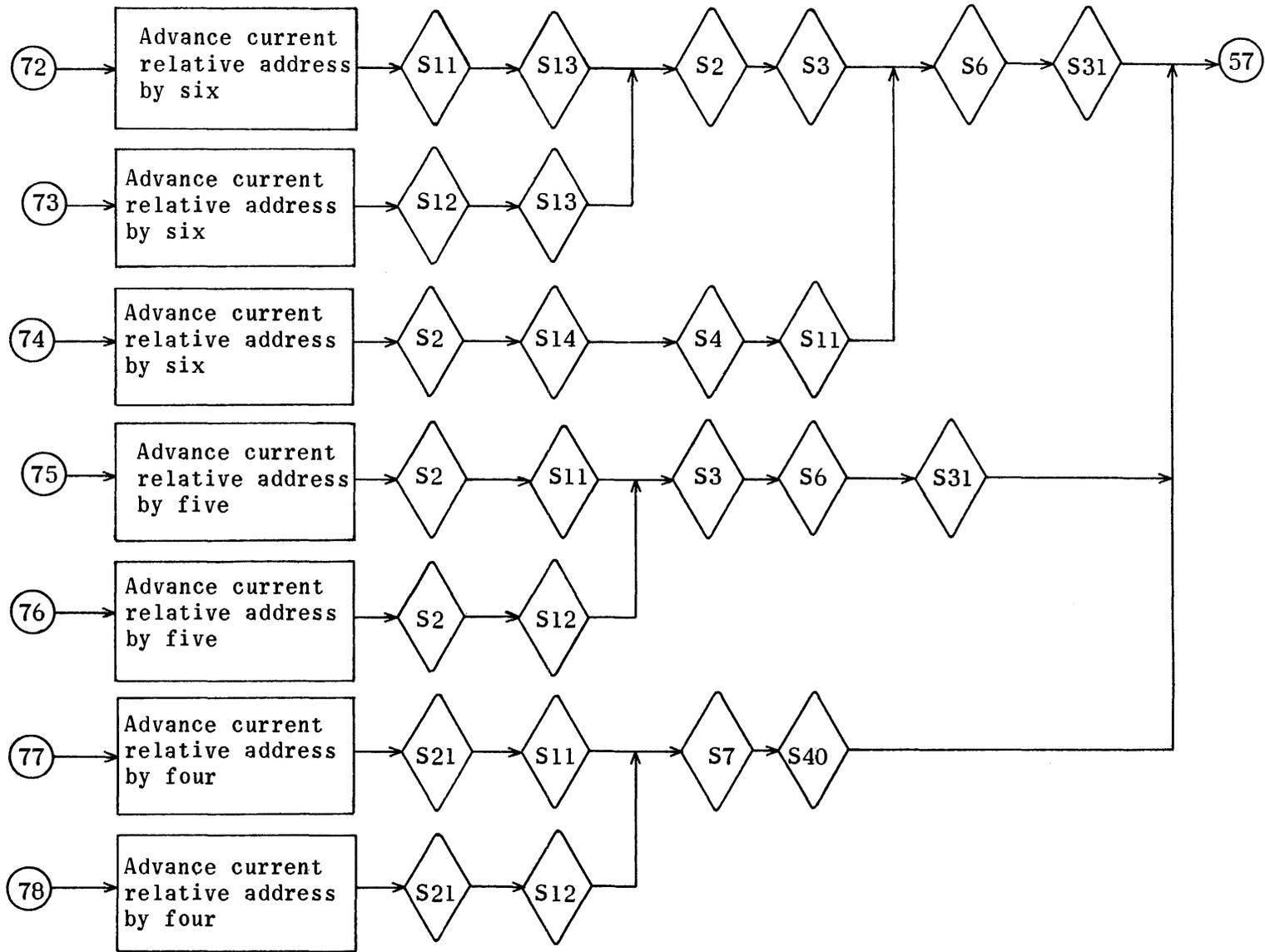


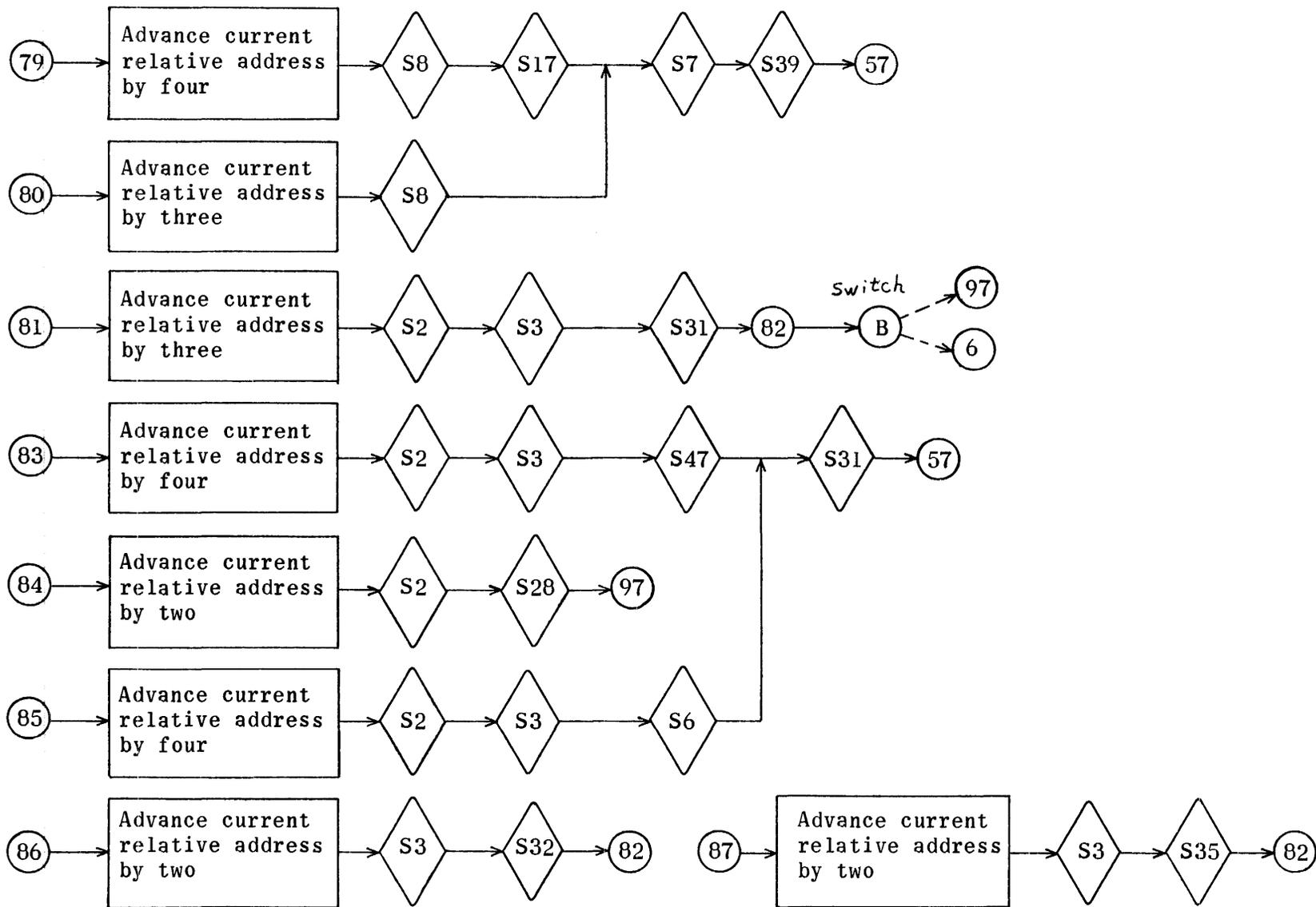


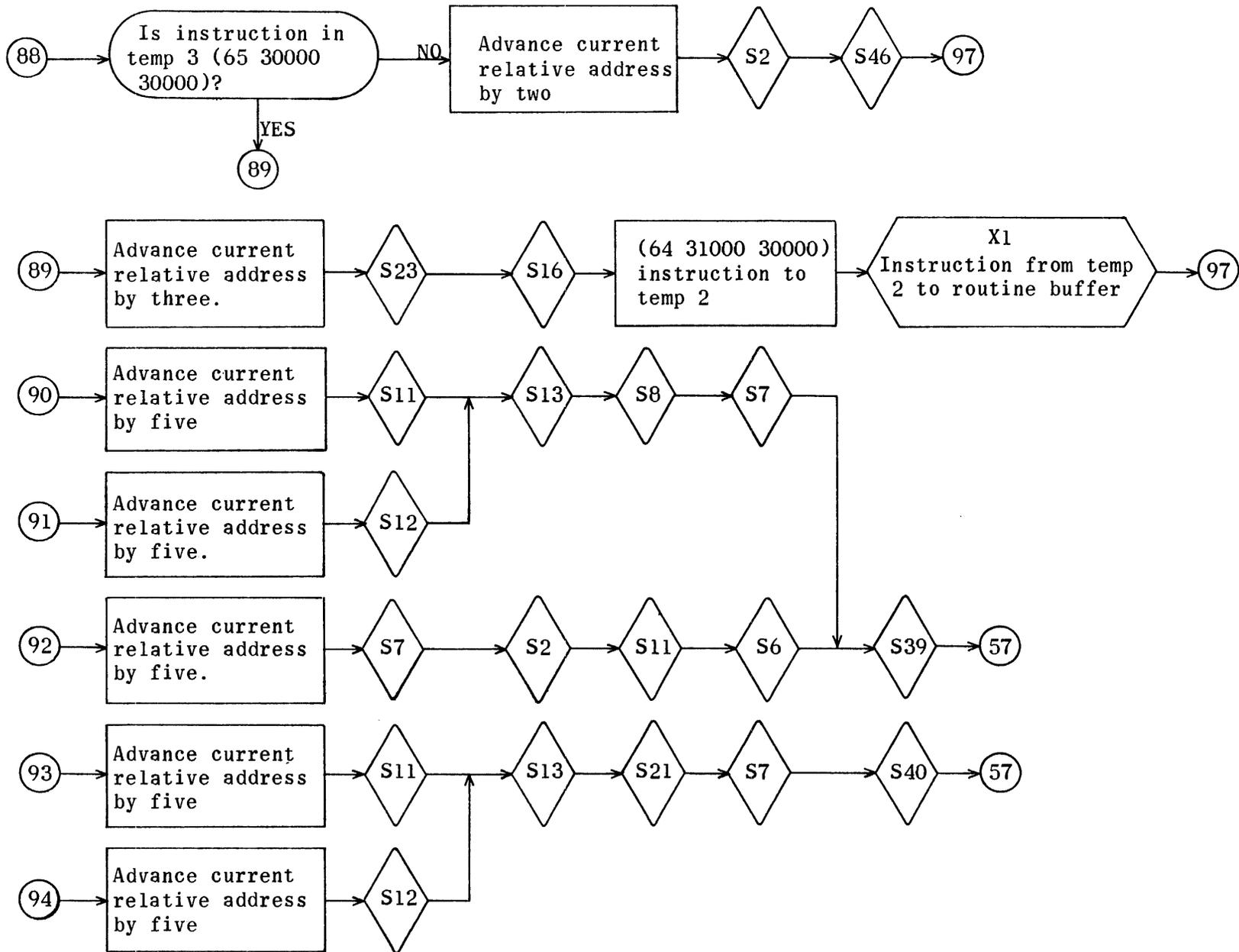


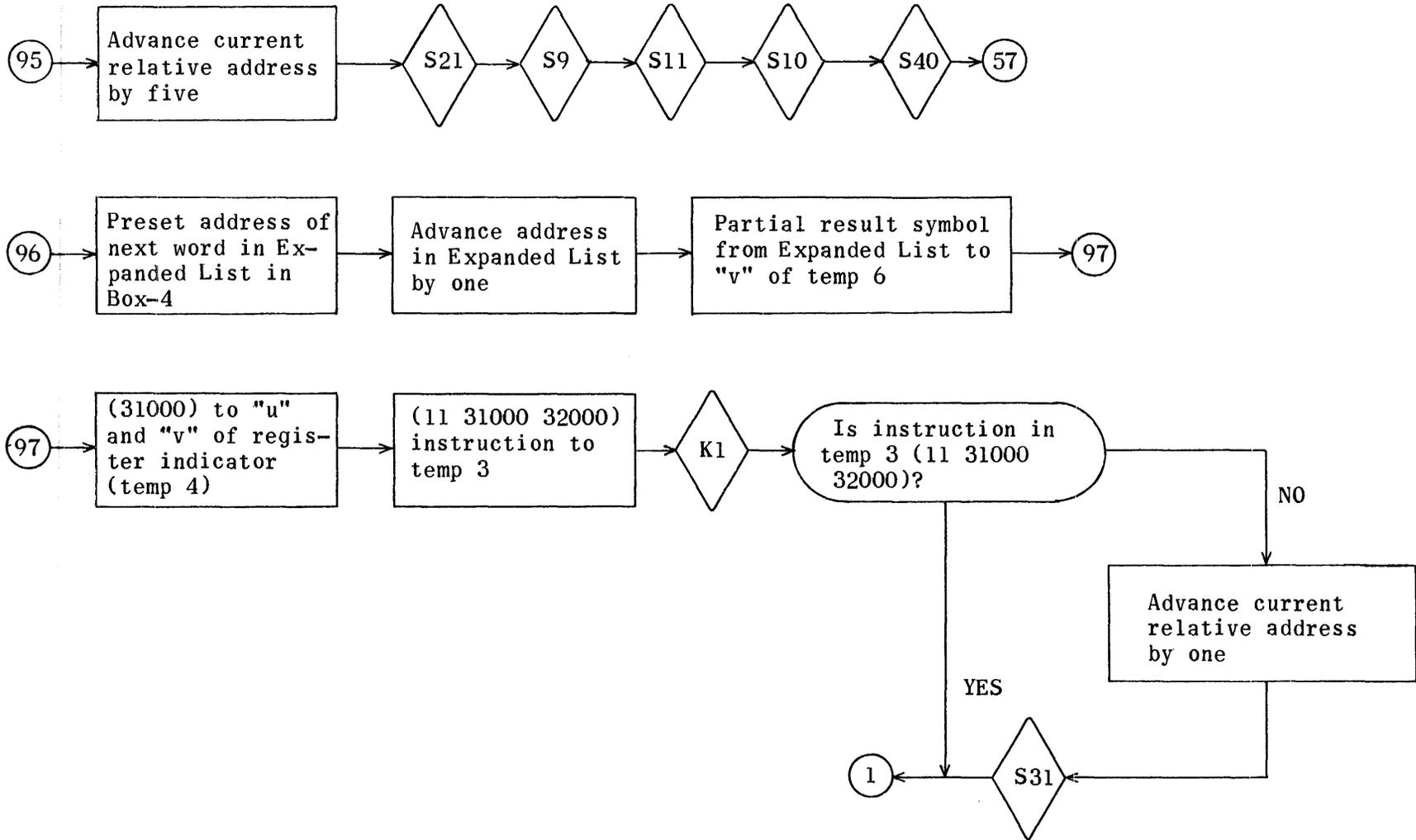




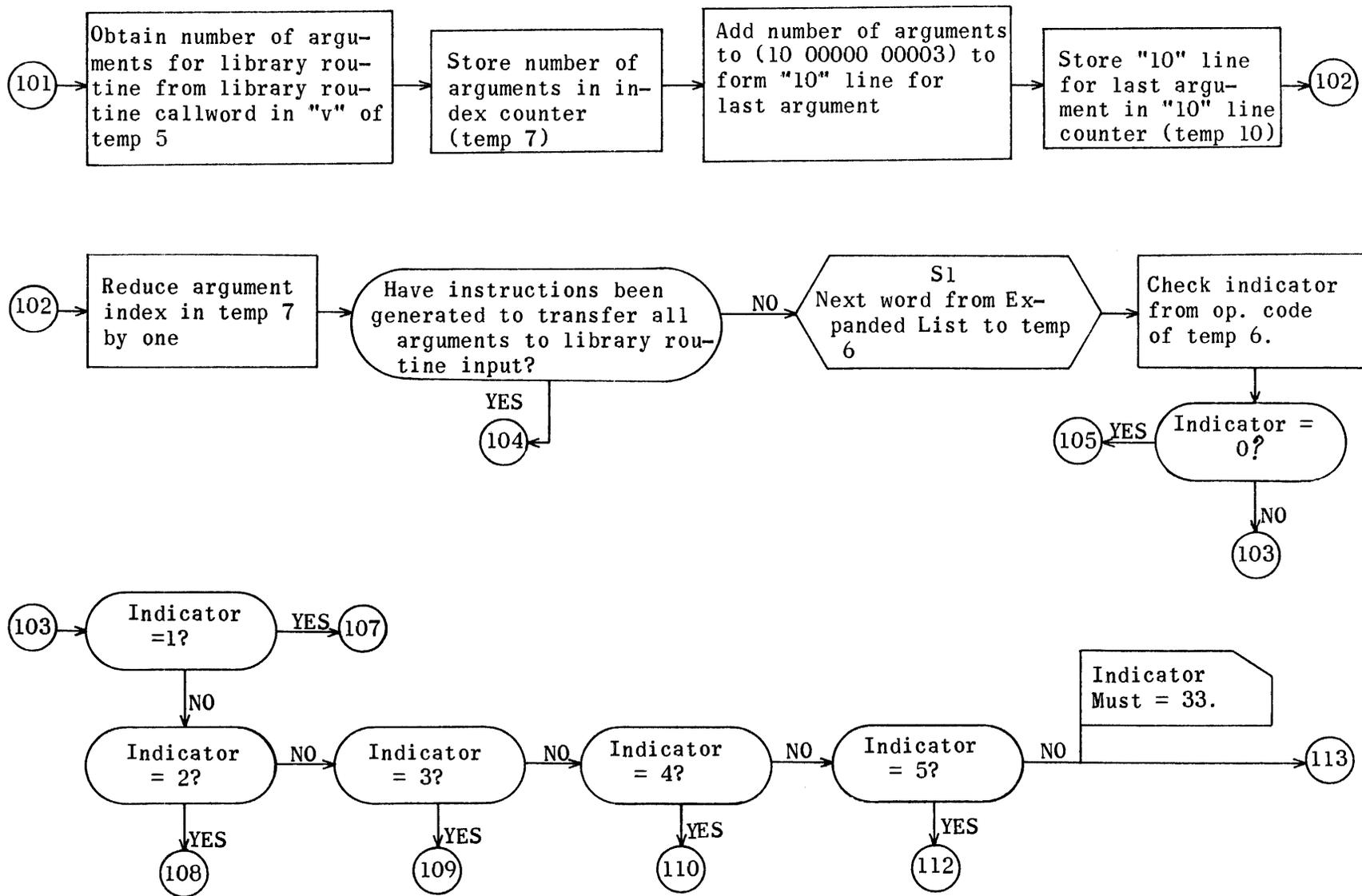


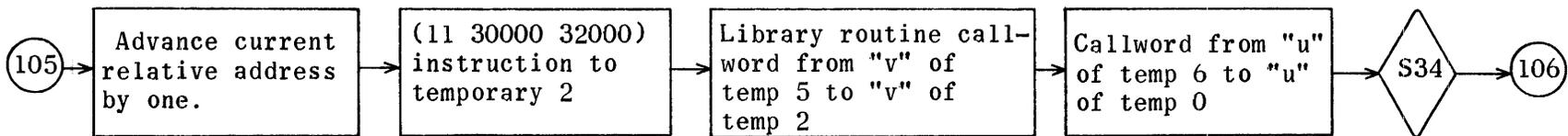
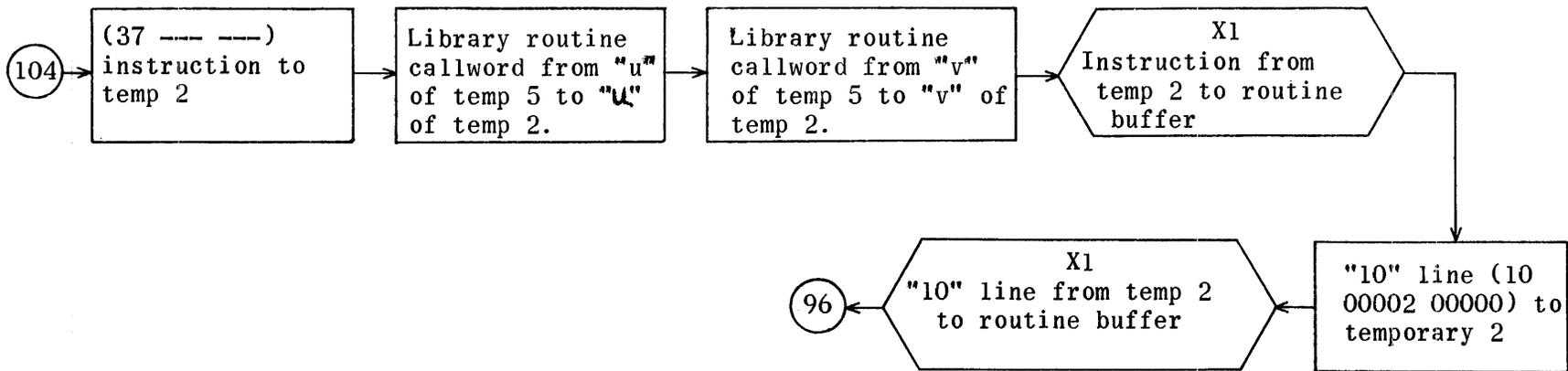


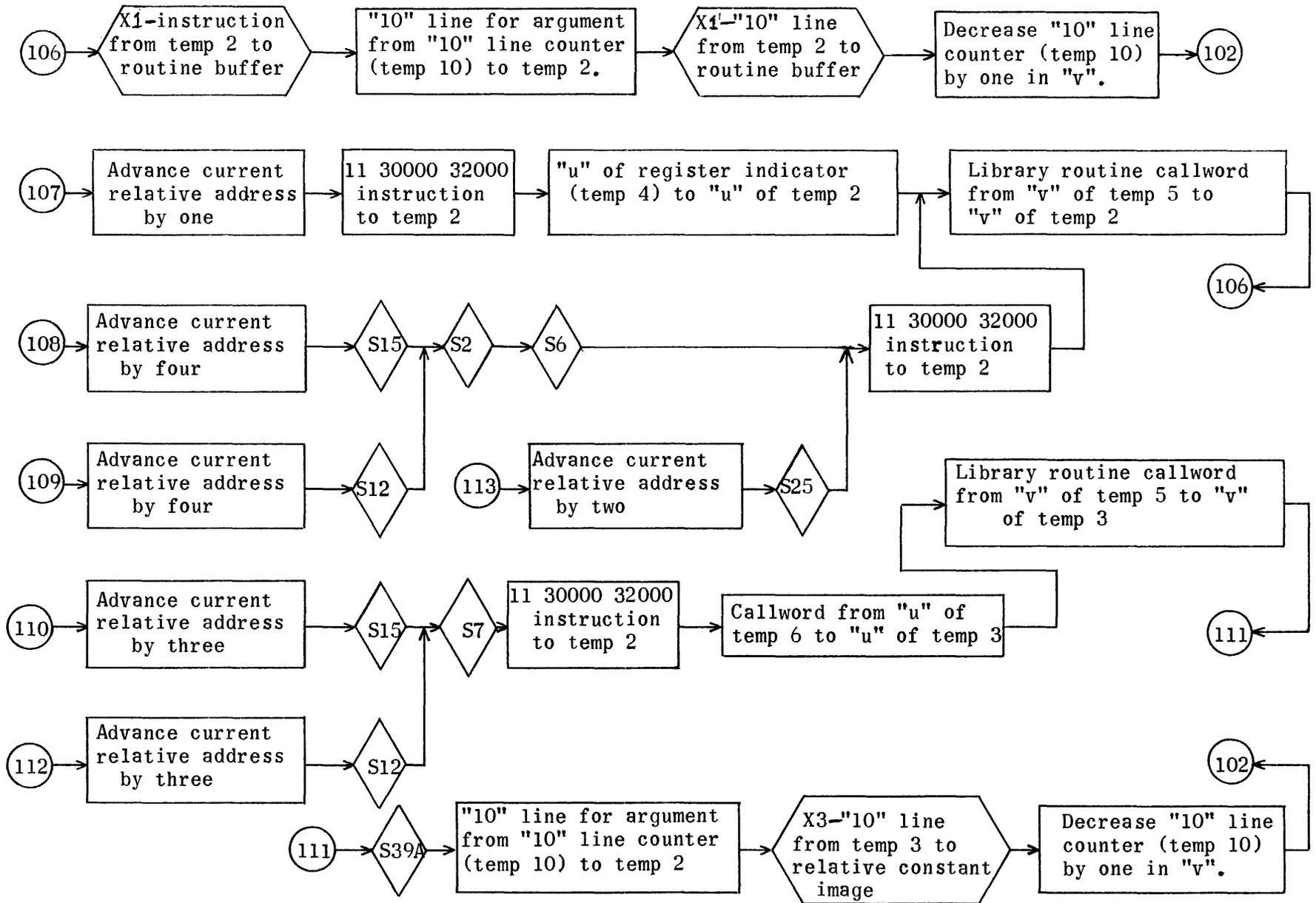




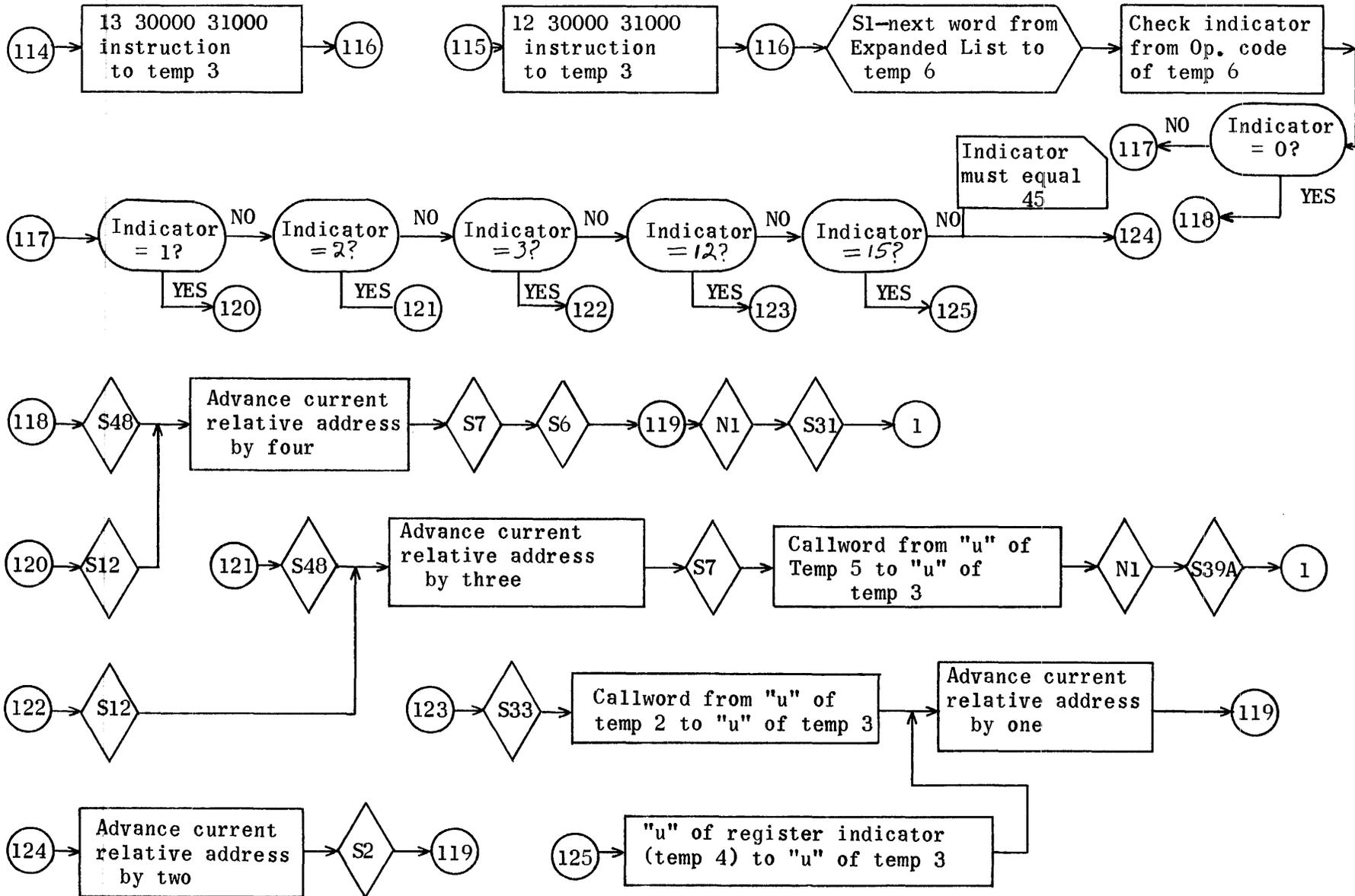
Equation Generation Phase (Library Routine Operator)





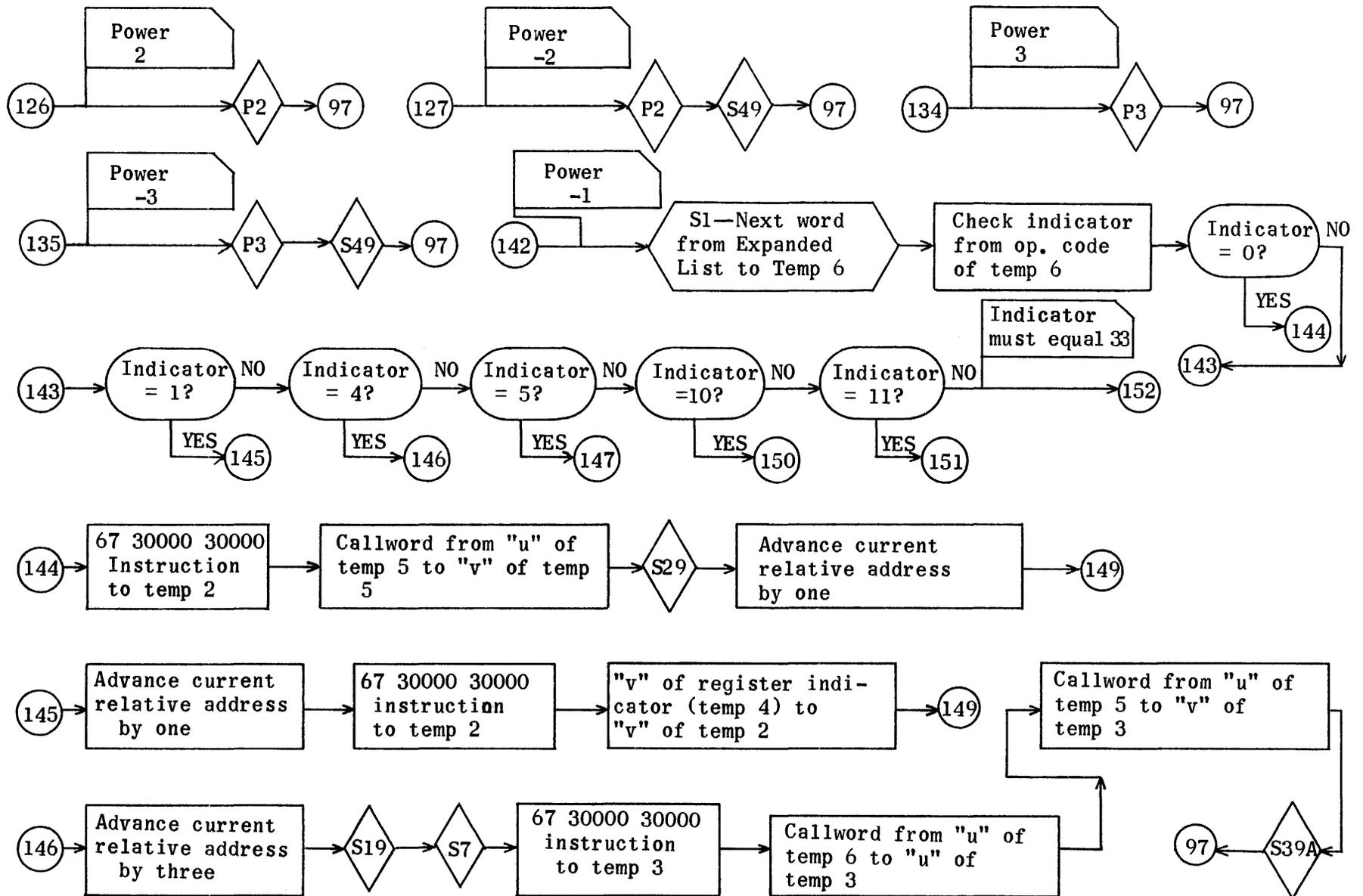


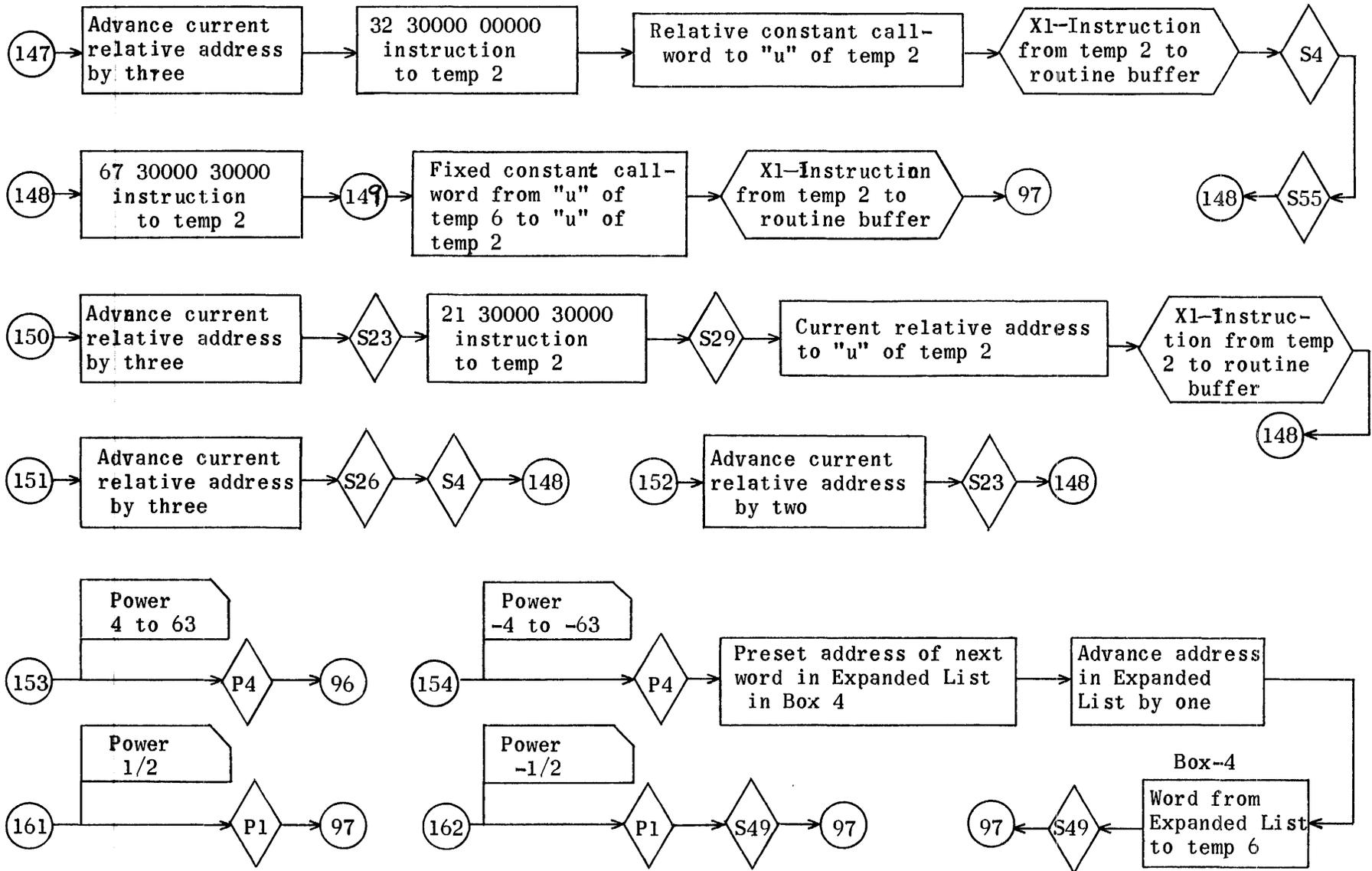
Equation Generation Phase (Floating Point Unary Minus and Absolute Value Operators)



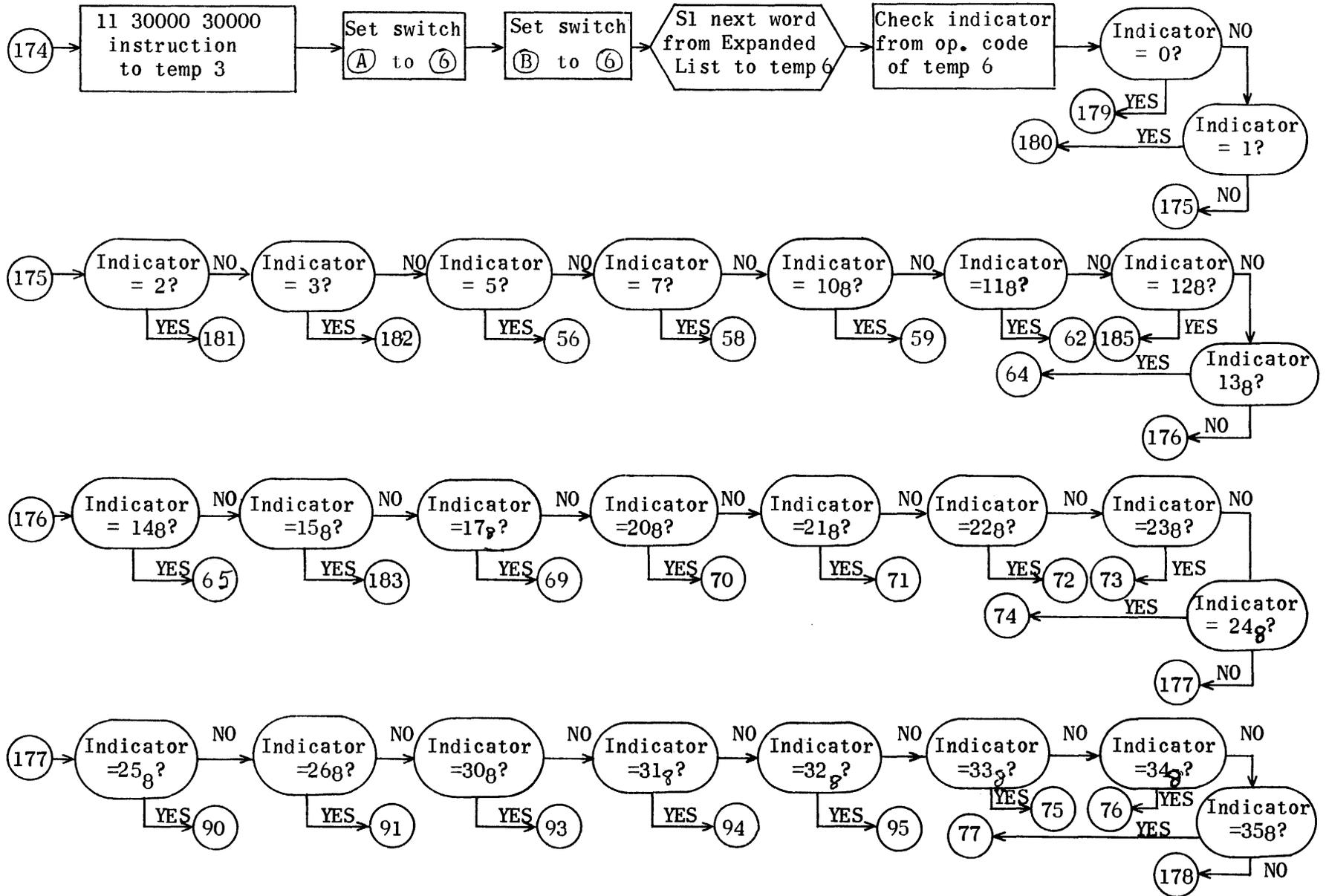
Equation Generation Phase (Integral Power Operators)

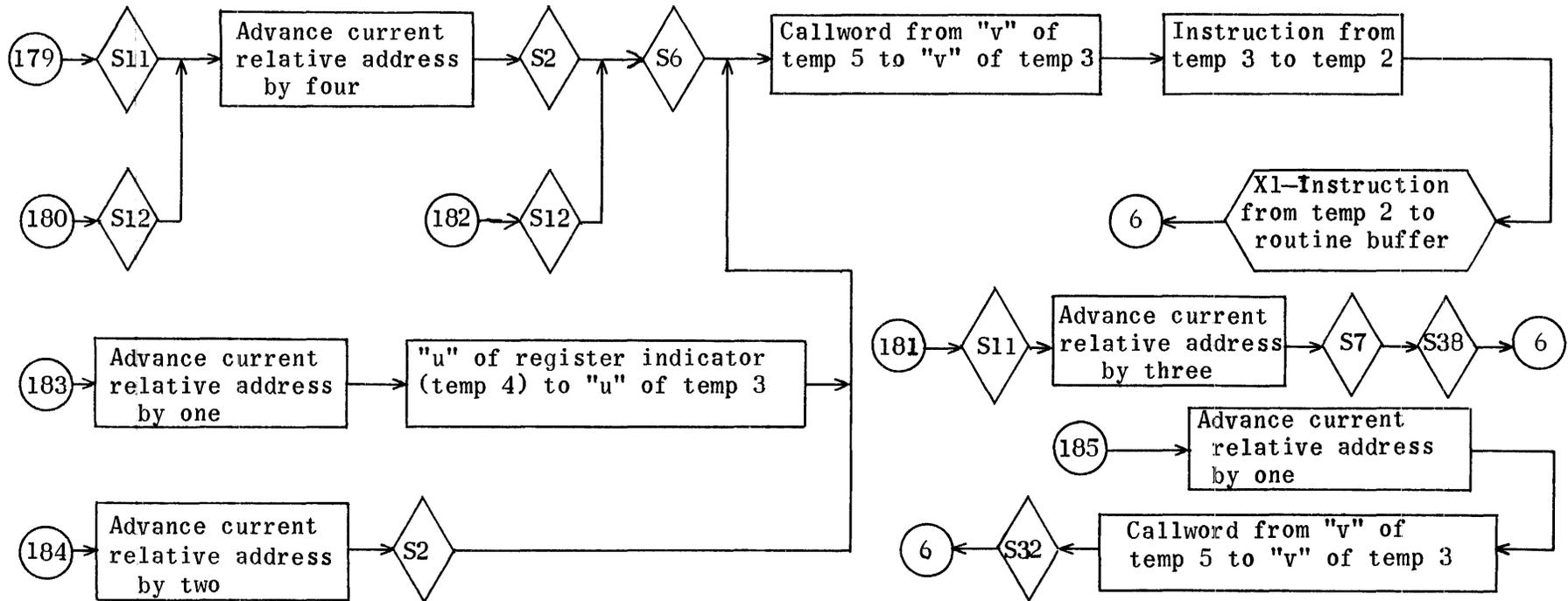
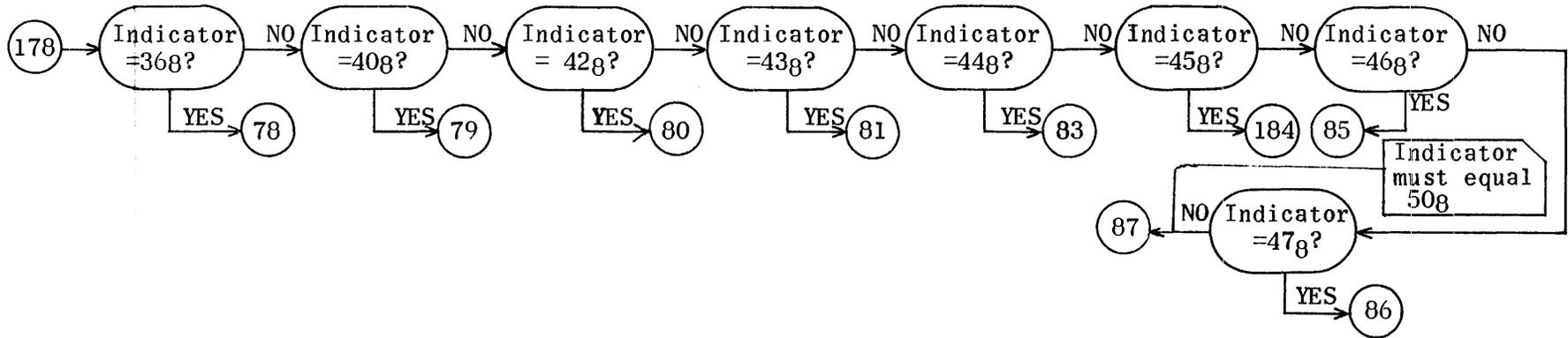
1378



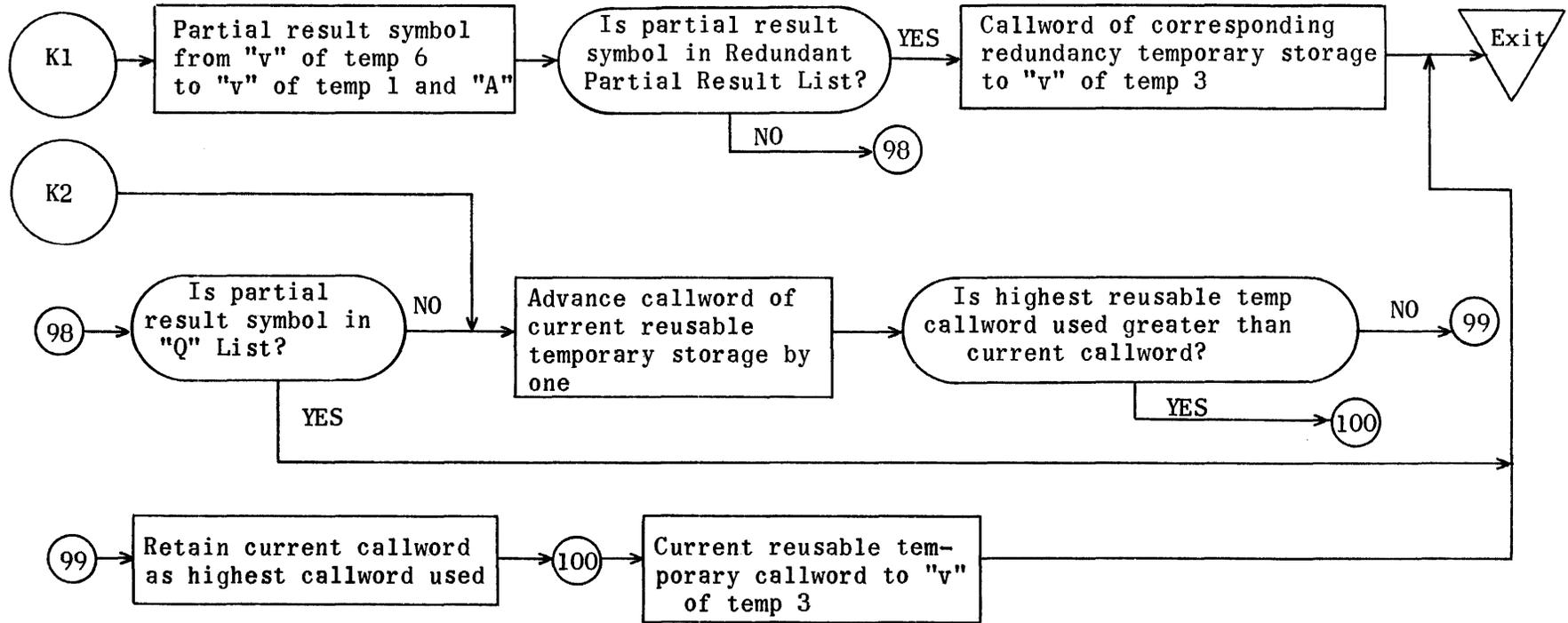


Equation Generation Phase (Operator to Store Result)



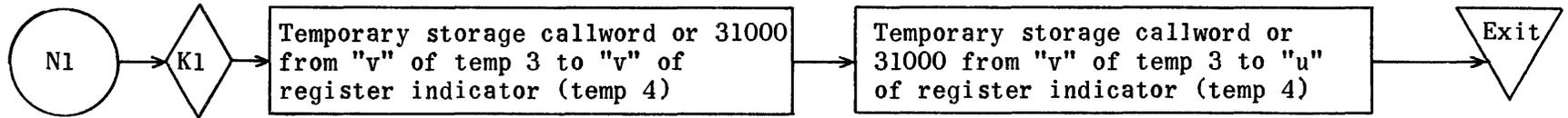


Equation Generation Subroutine for Floating Point Operators

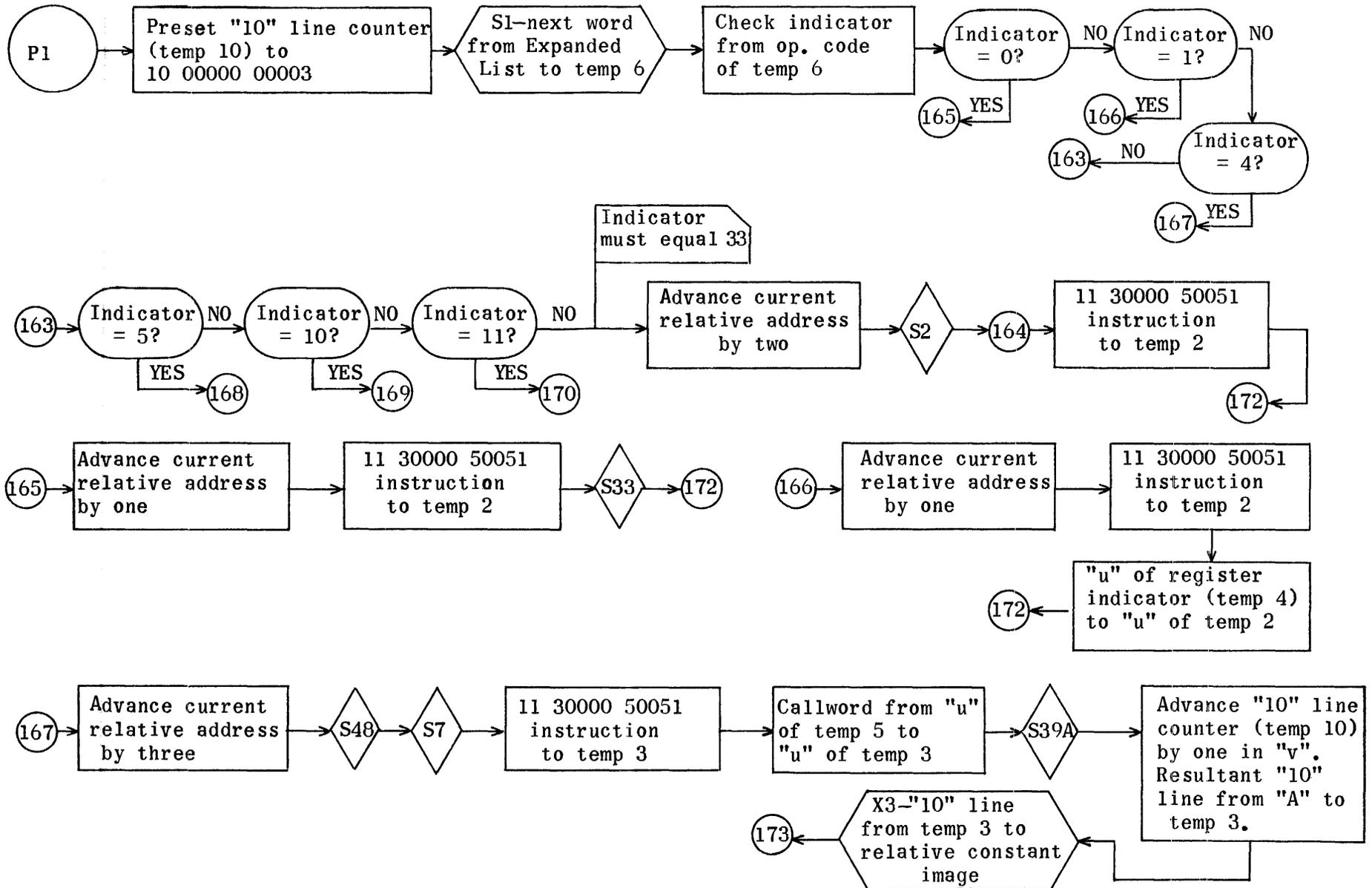


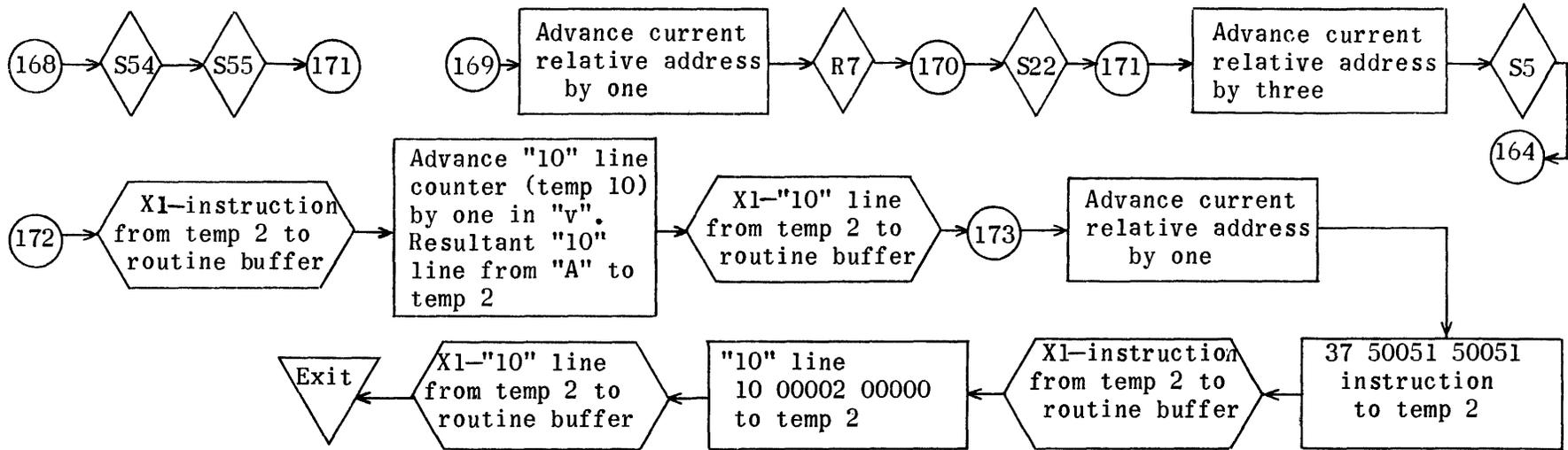
1382

Equation Generation Subroutine for Floating Point Unary Minus and Absolute Value Operators

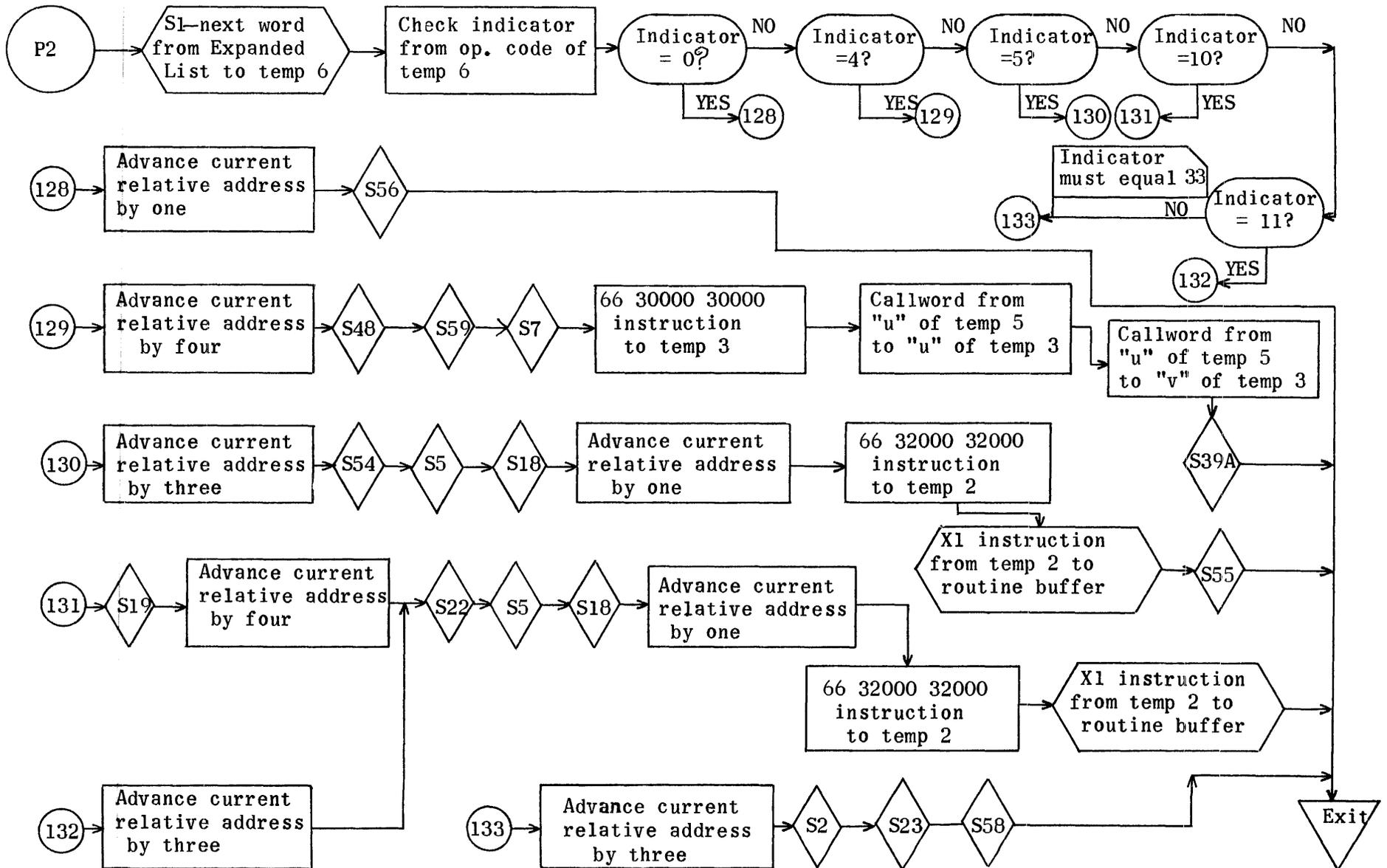


Equation Generation Subroutine for Power (1/2) and (-1/2)

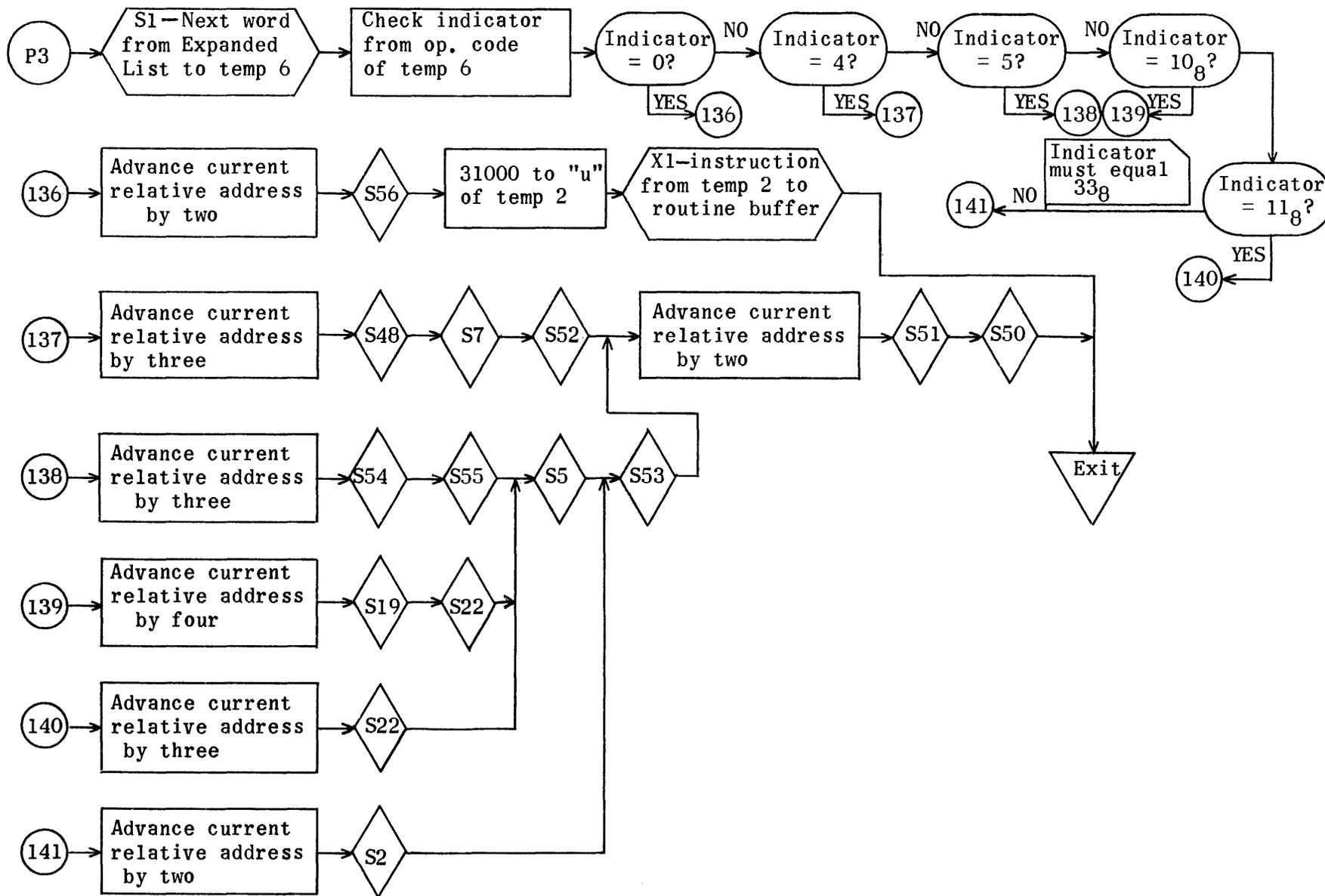




Equation Generation Subroutine for Power (2) and (-2)

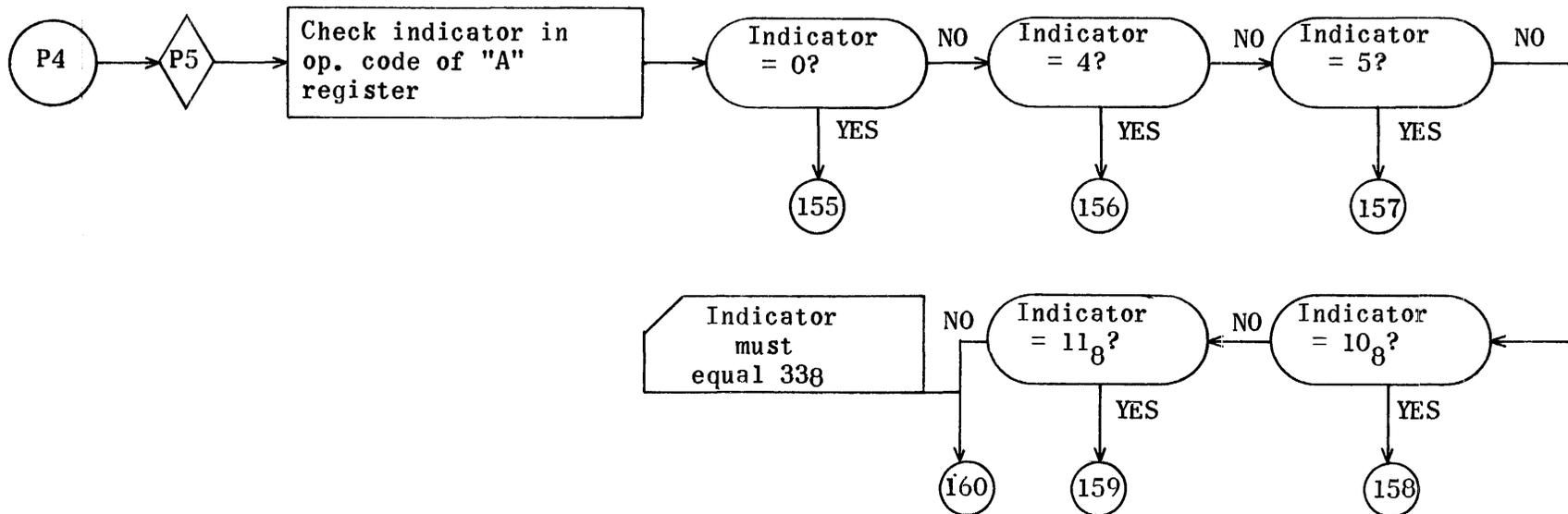
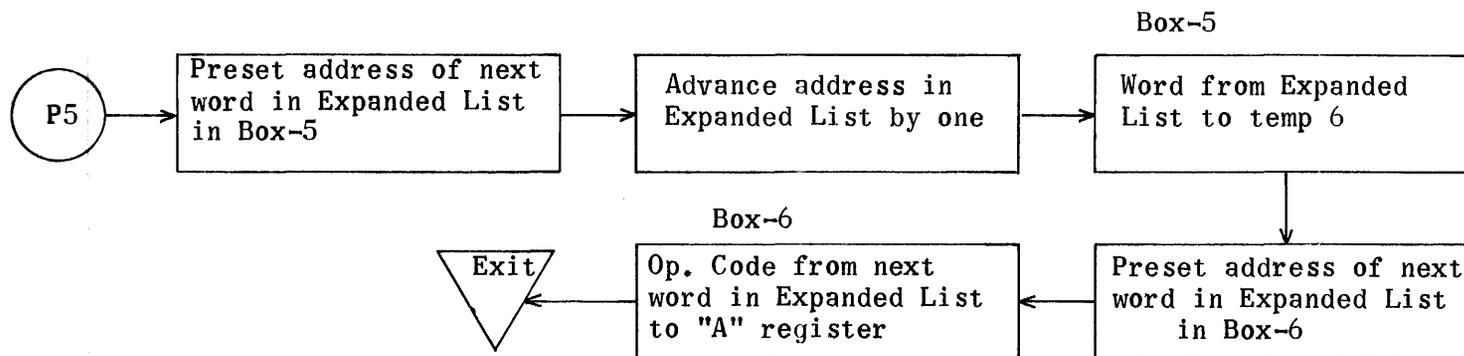


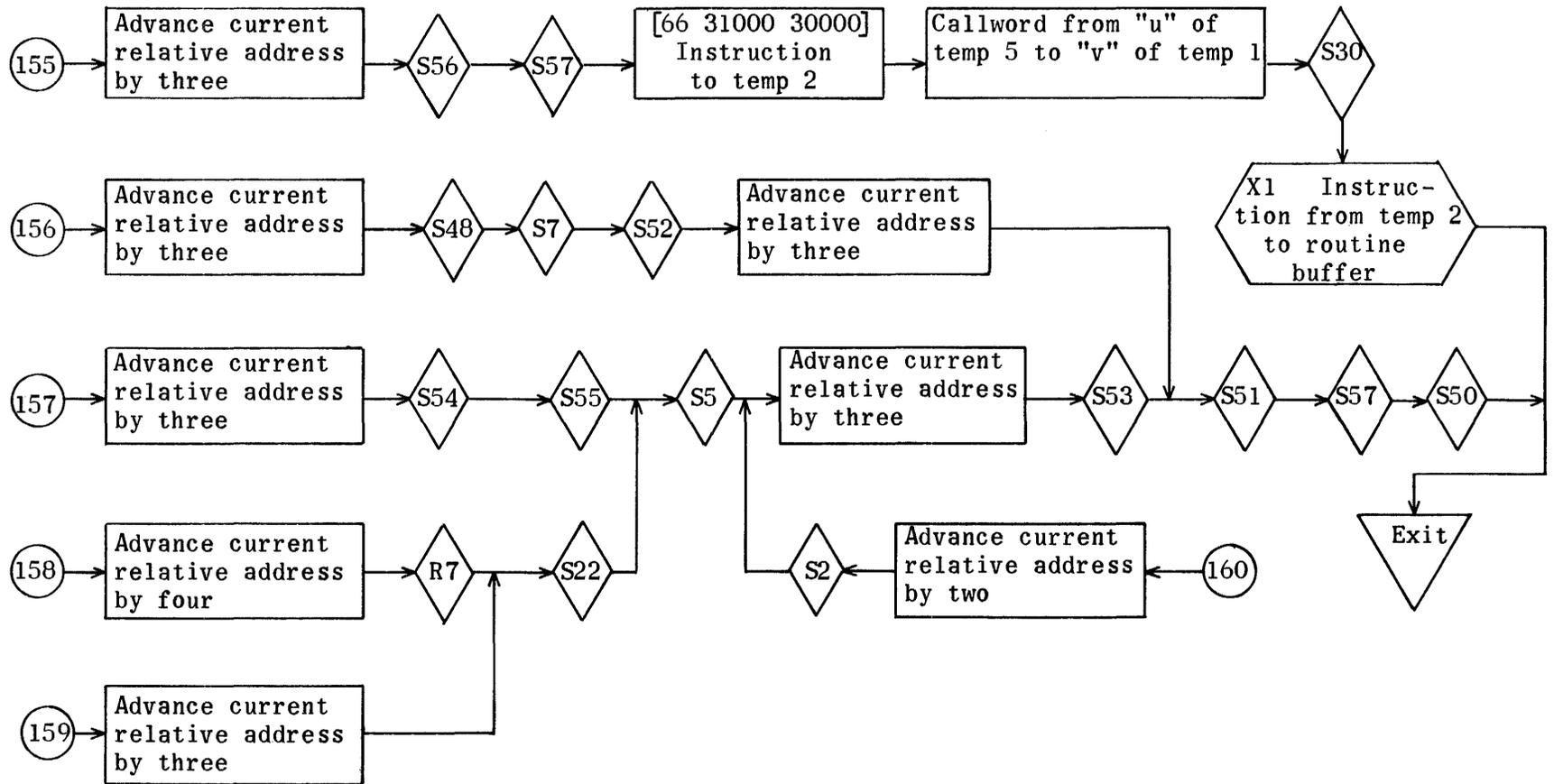
Equation Generation Subroutine for Power (3) and (-3)



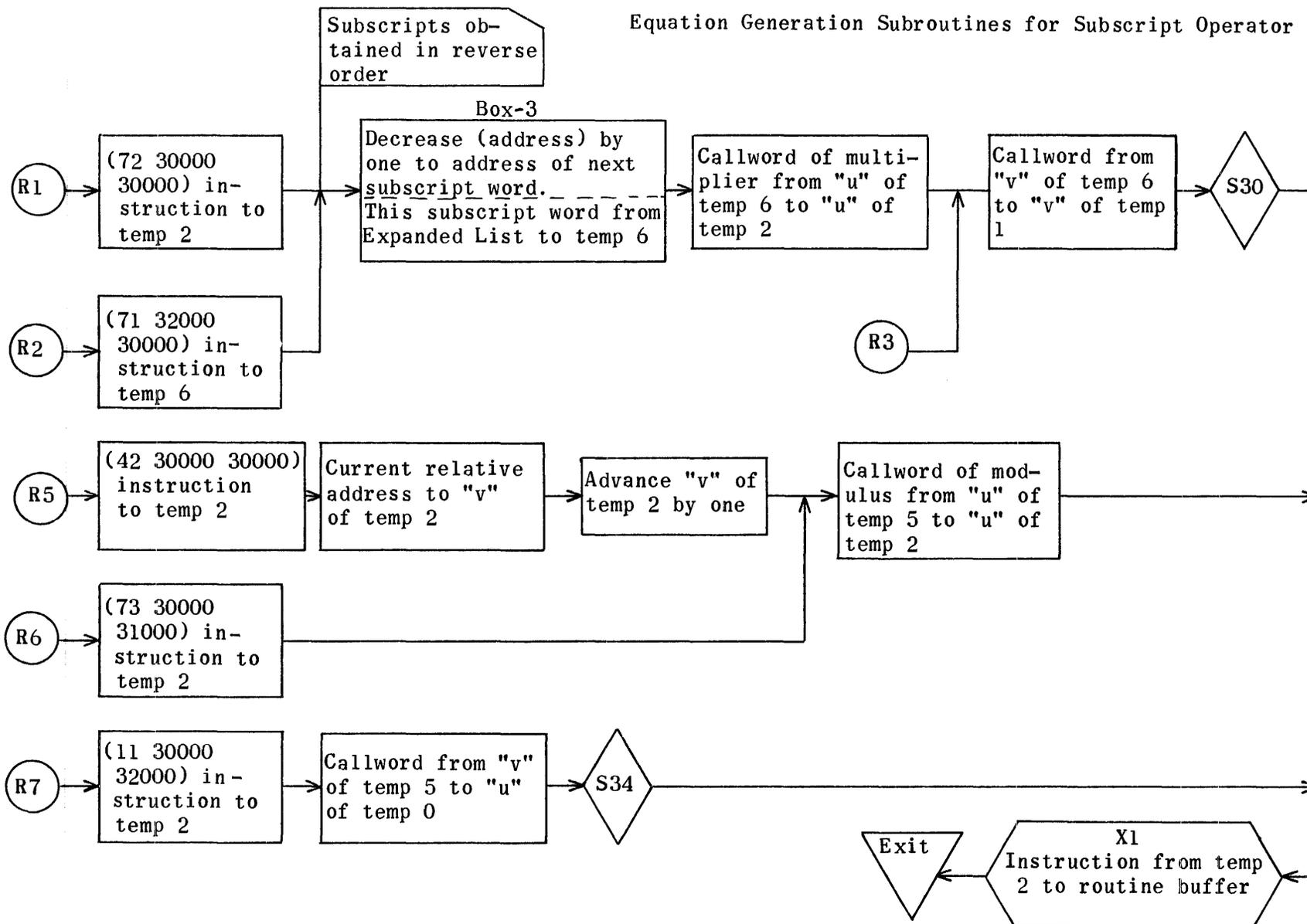
1386

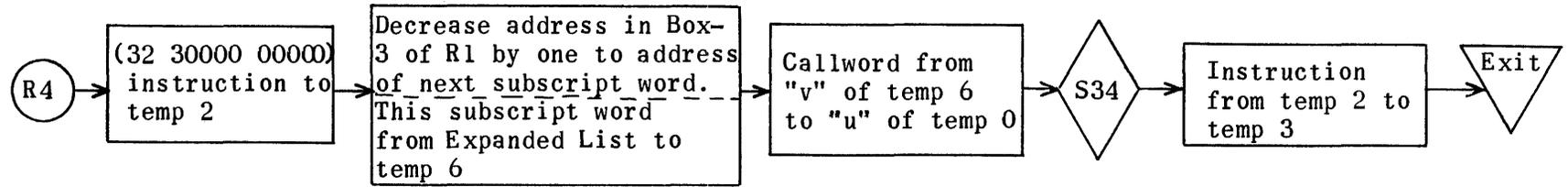
Equation Generation Subroutines for Power (4 to 63) and (-4 to -63)



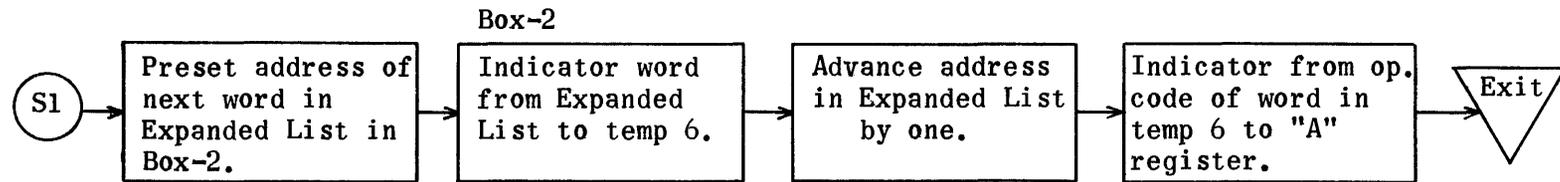


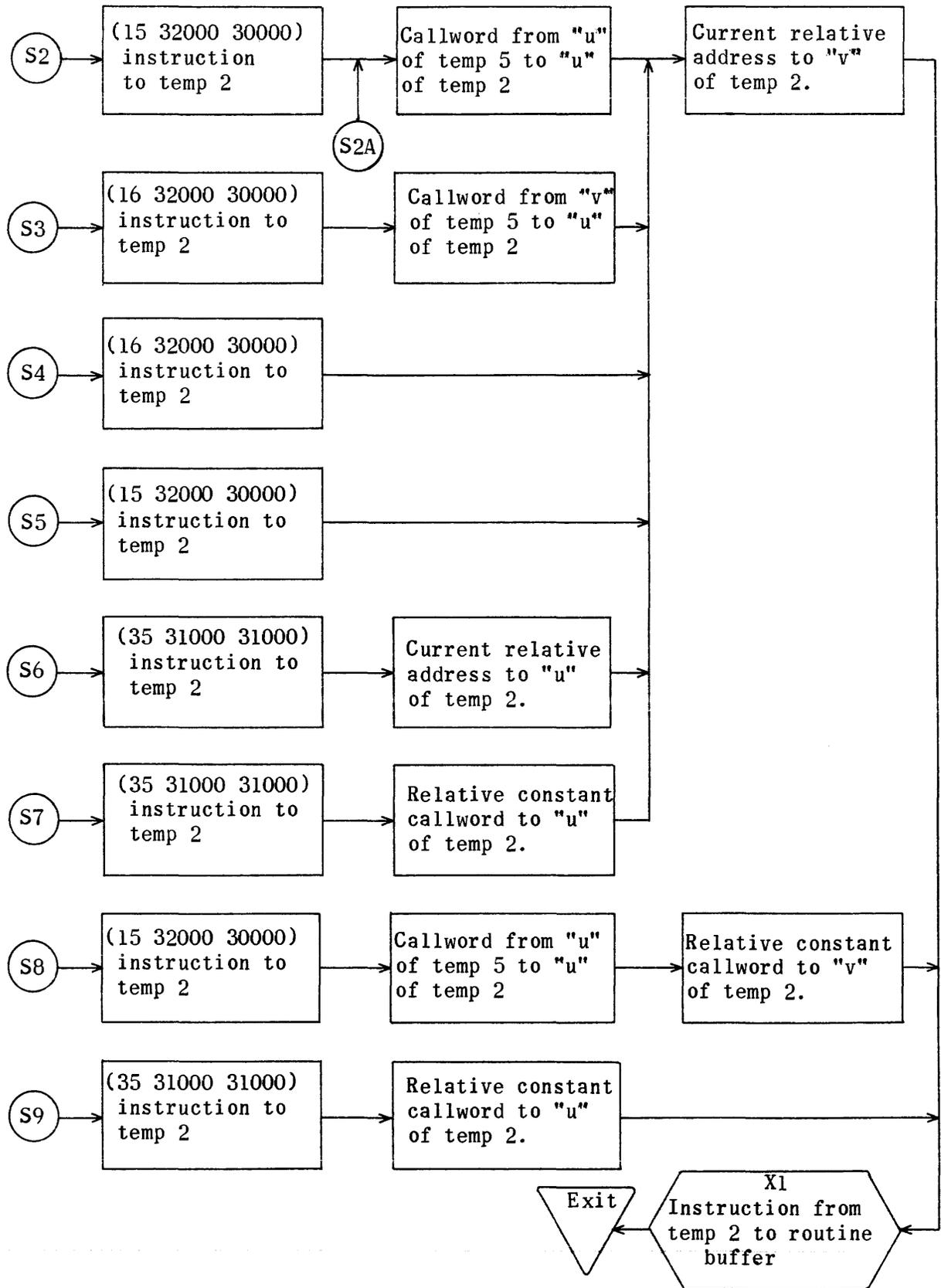
Equation Generation Subroutines for Subscript Operator

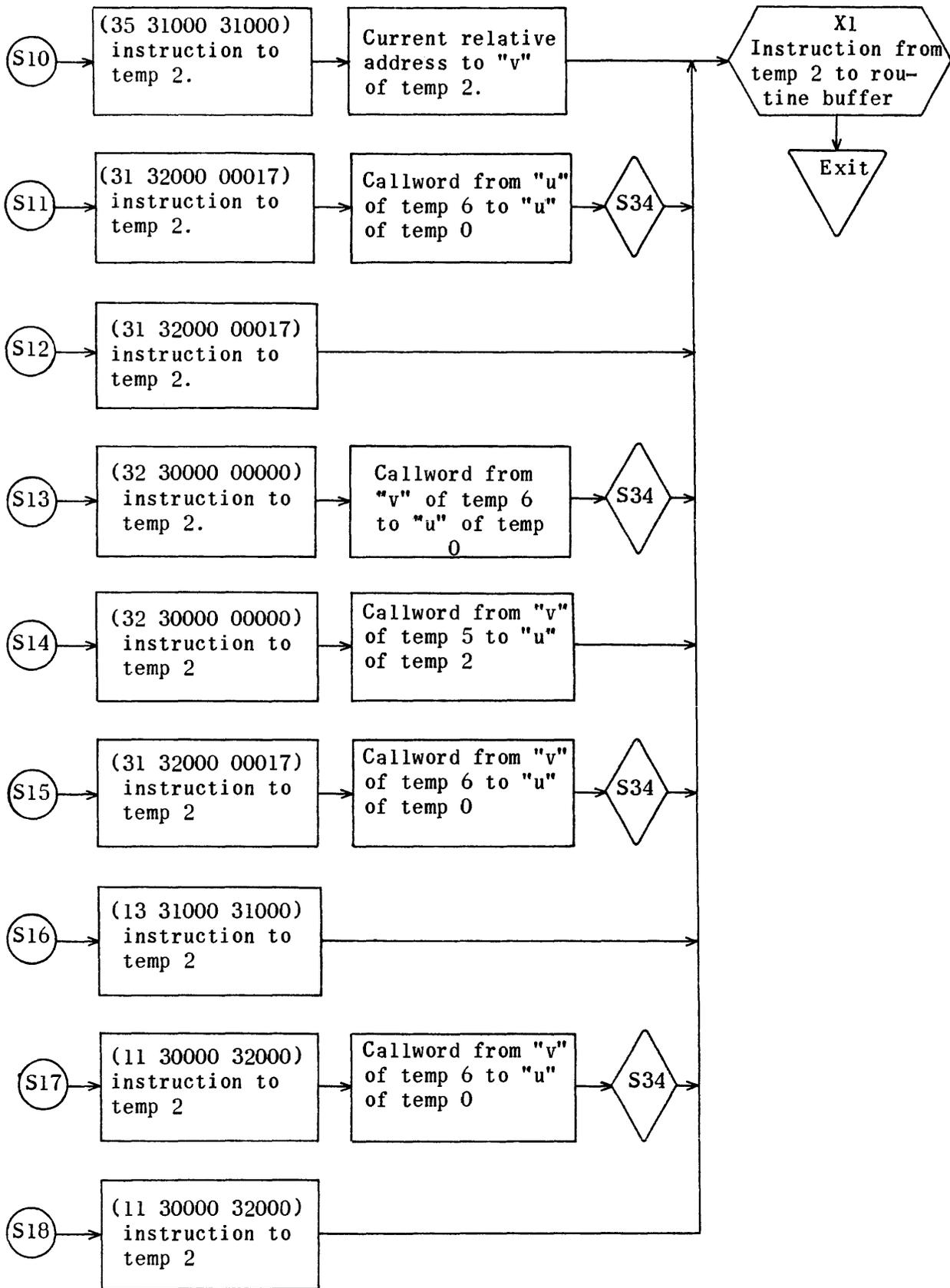


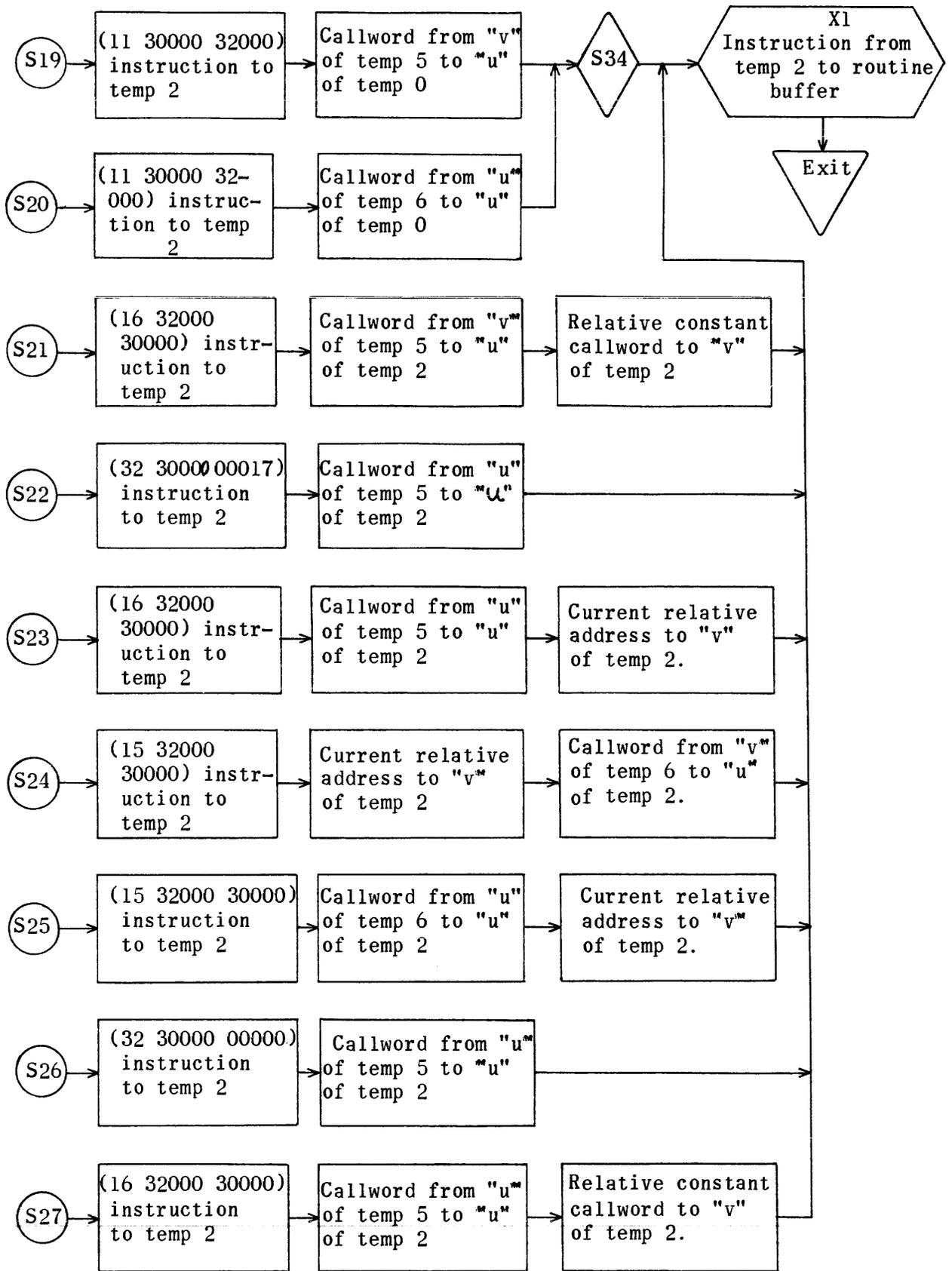


Equation Generation Subroutines

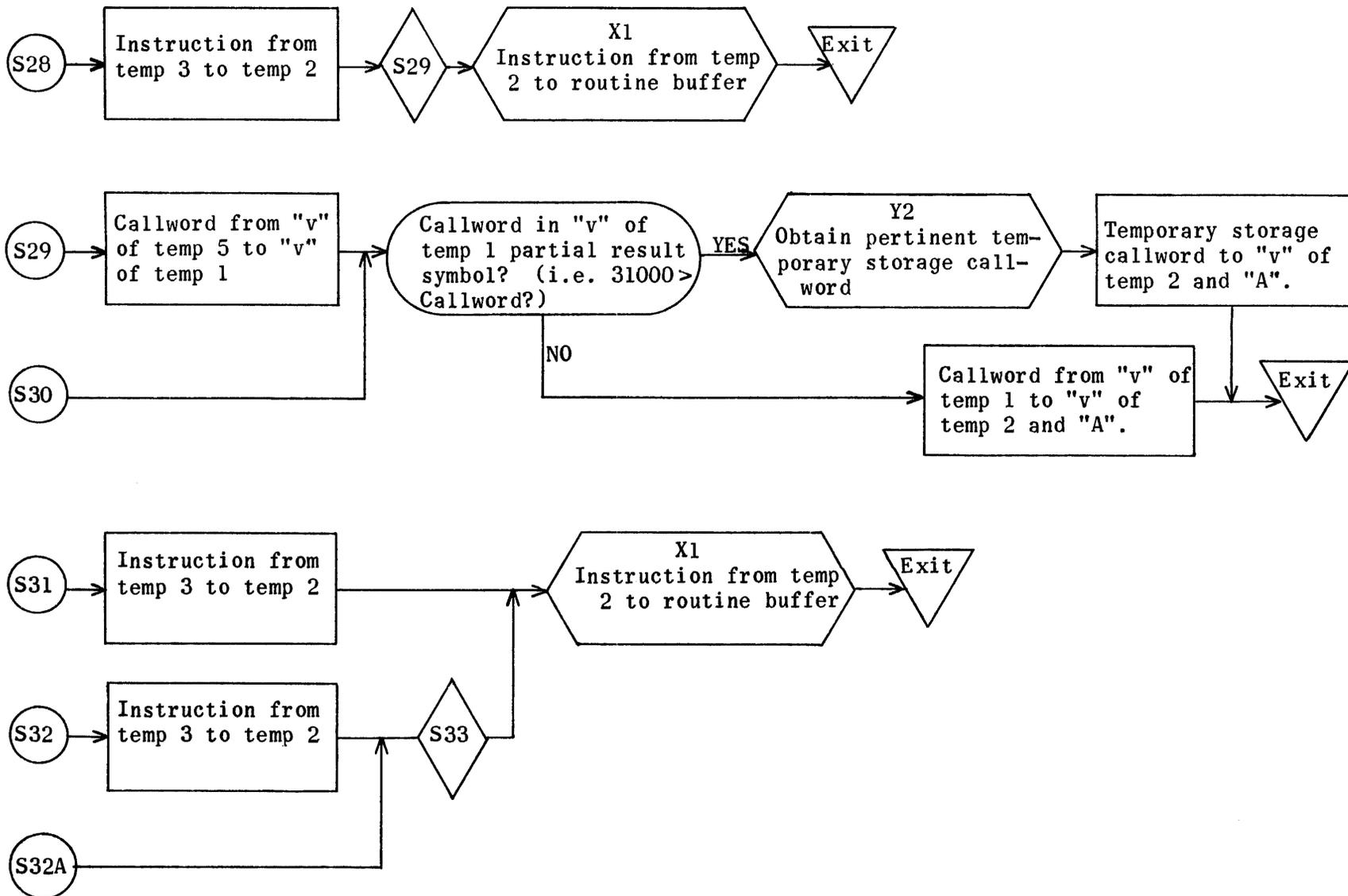


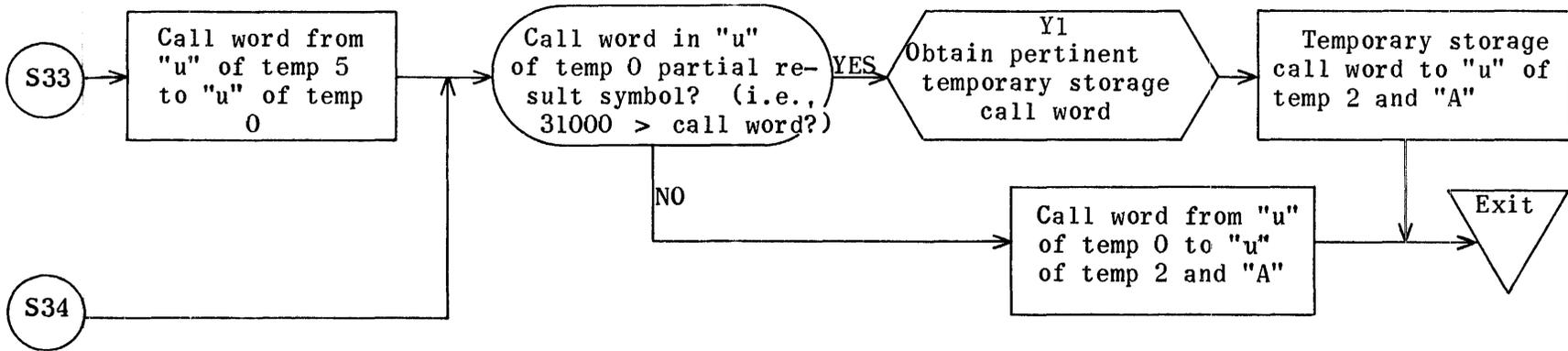




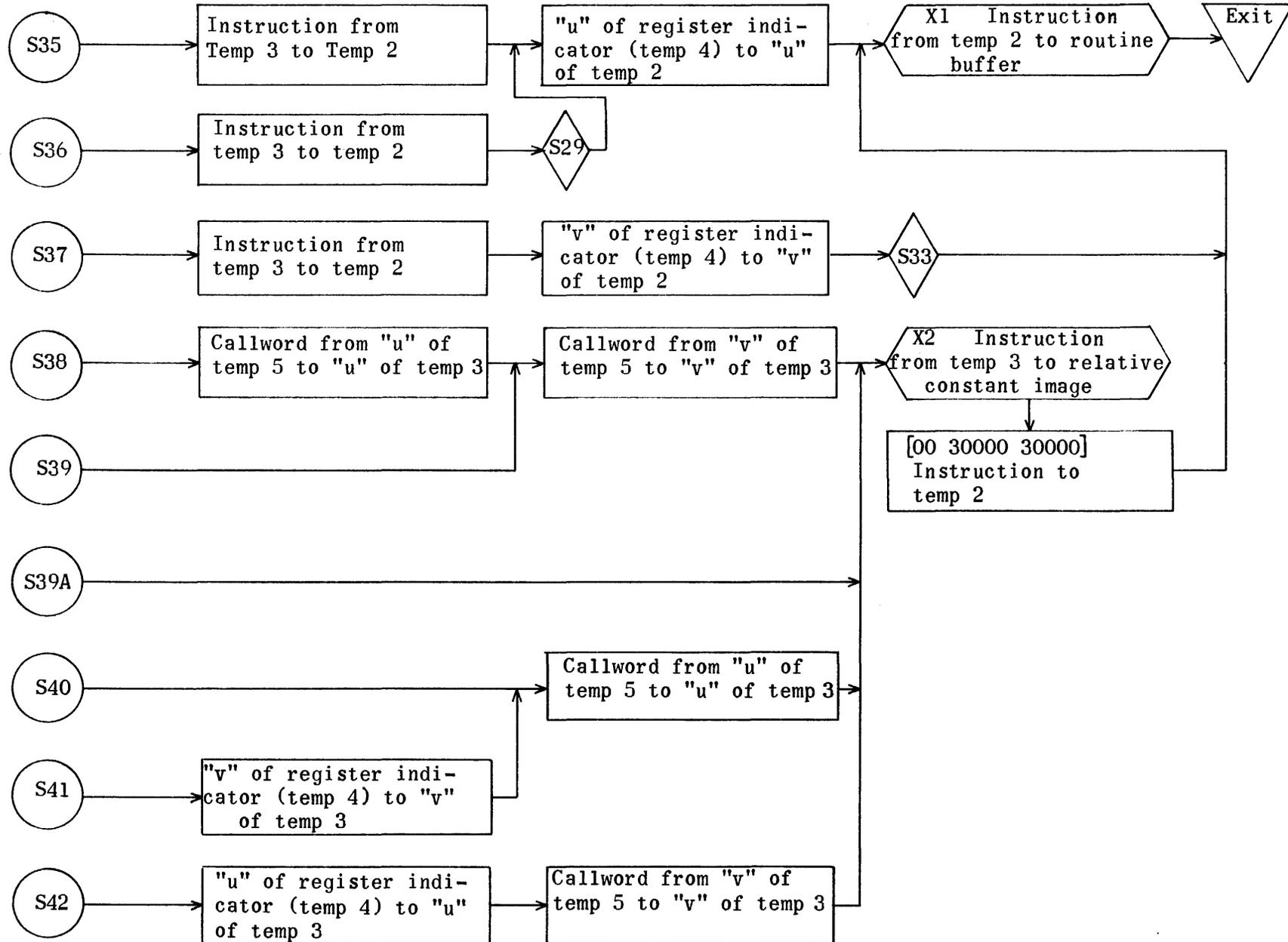


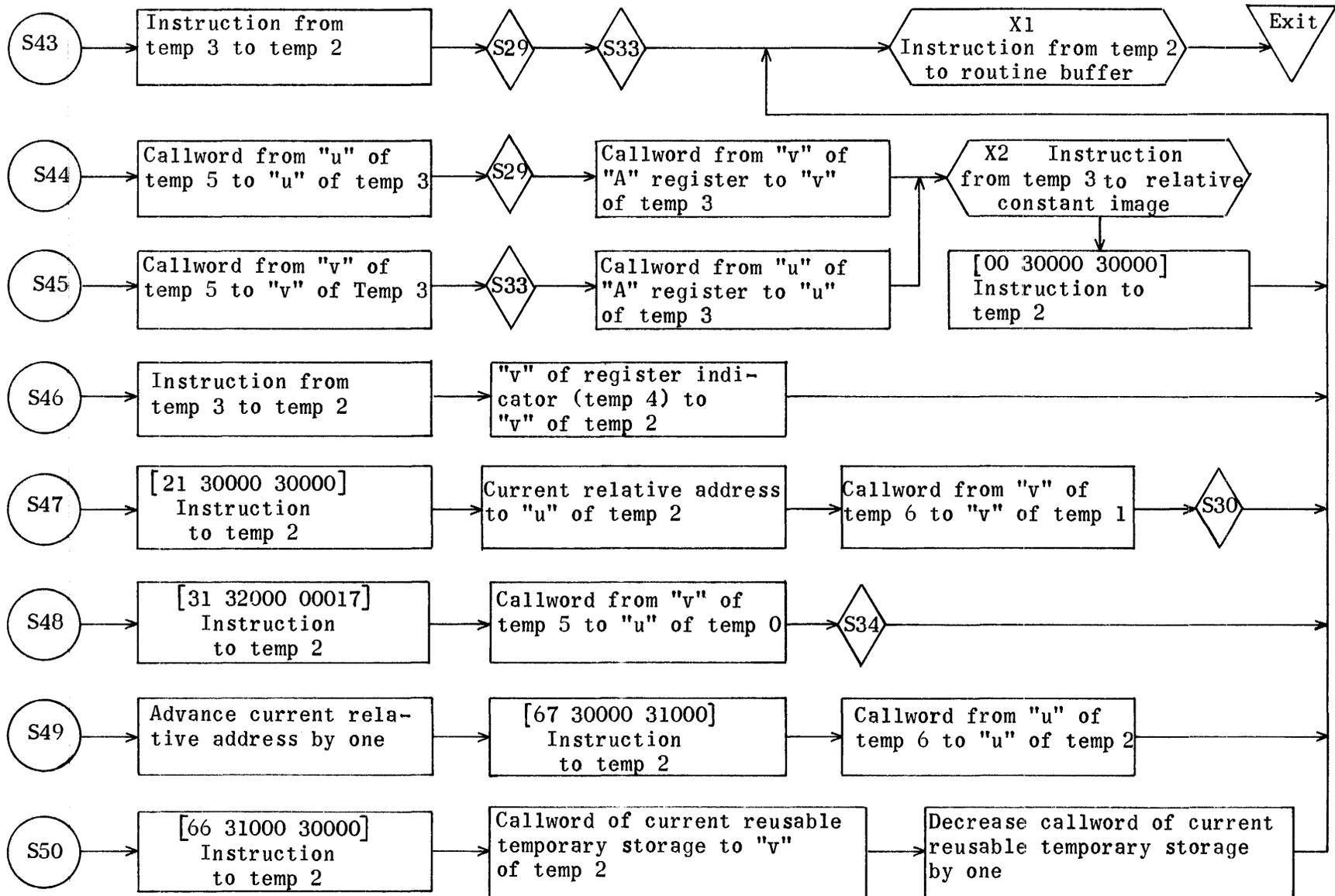
Equation Generation Subroutines

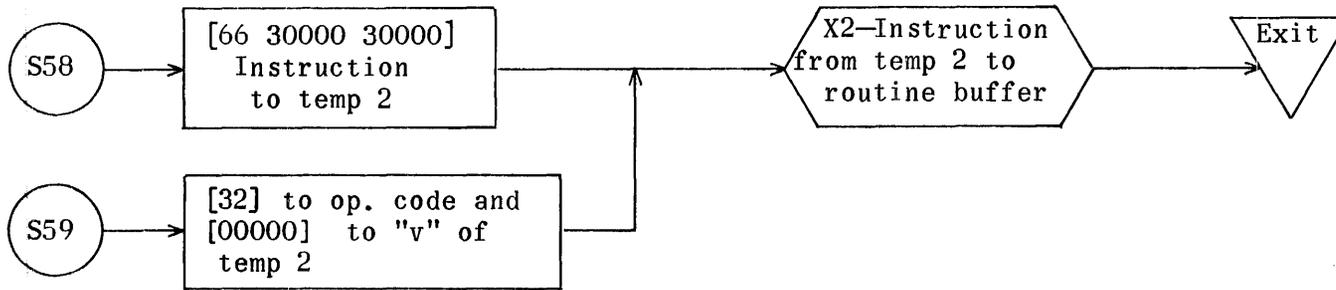




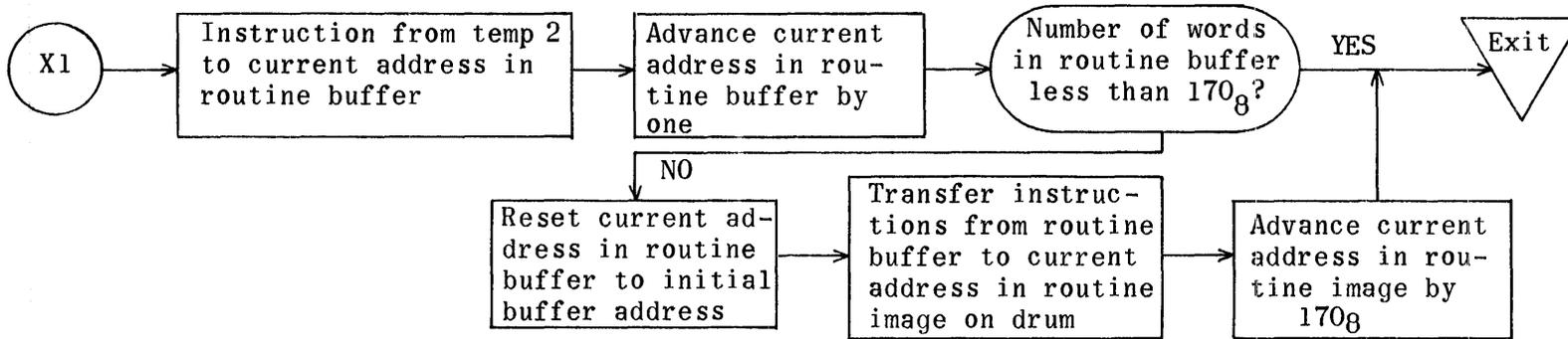
Equation Generation Subroutines



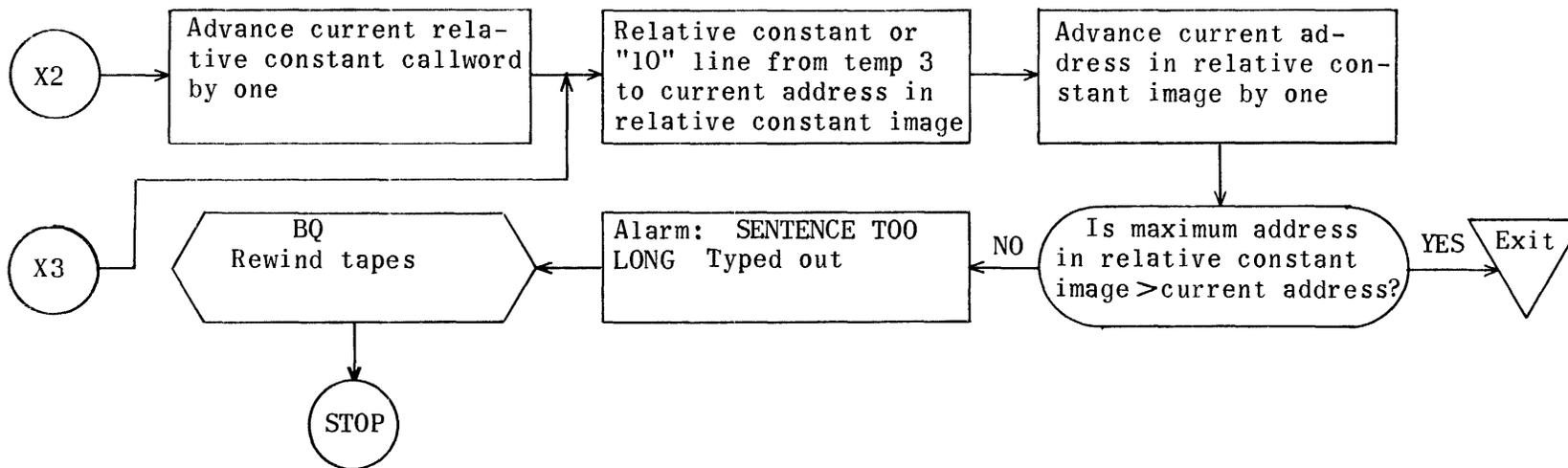




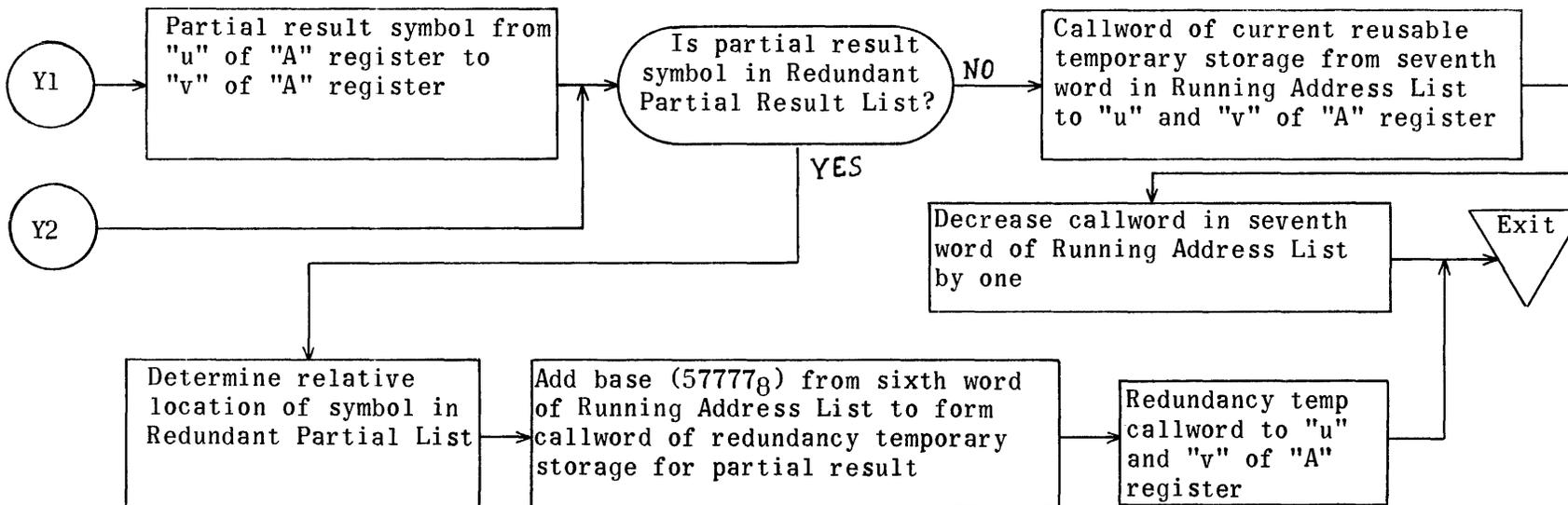
Equation Generation Subroutine to Store Instruction in Routine Buffer



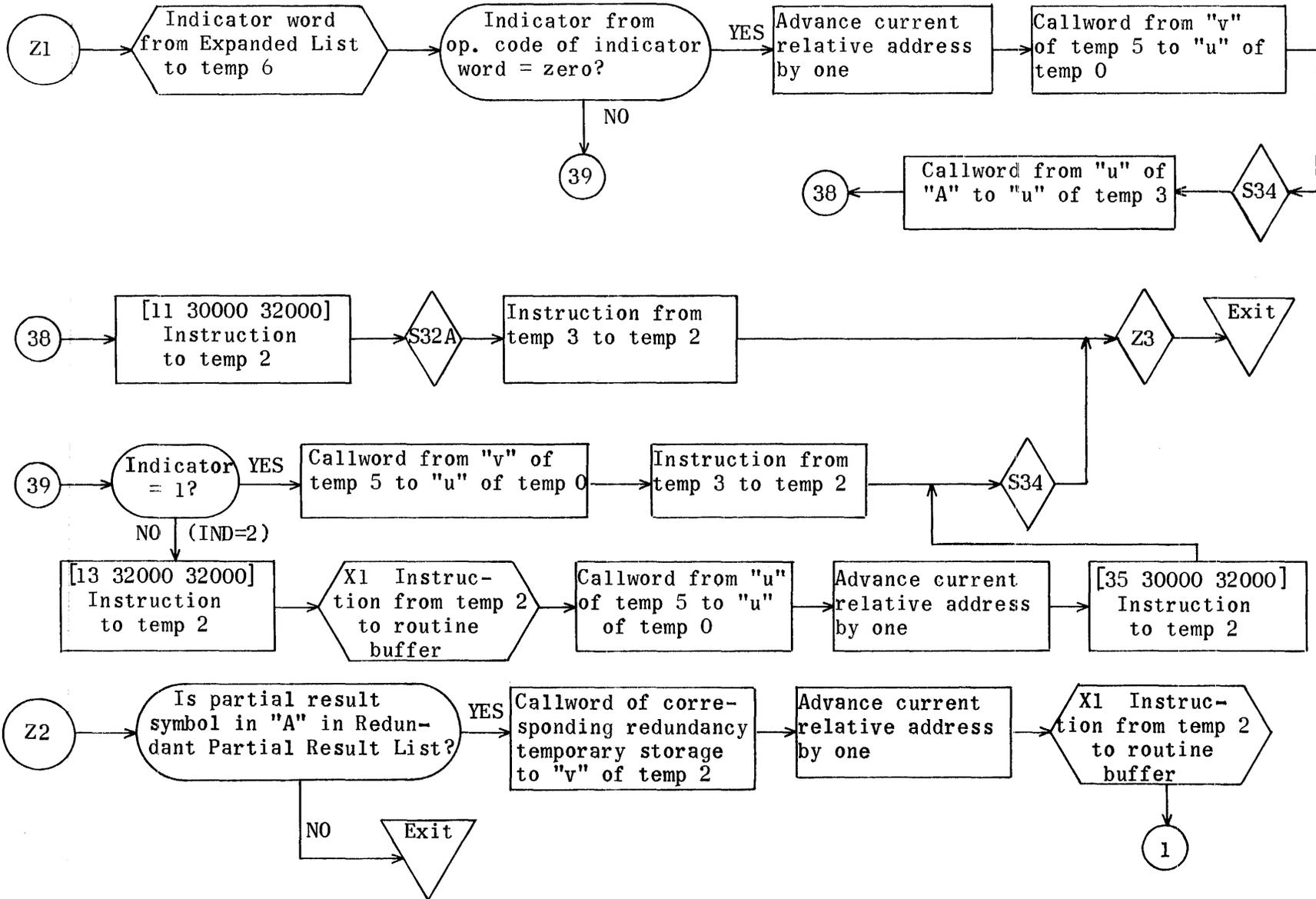
Equation Generation Subroutine to Store Relative Constant in Relative Constant Image



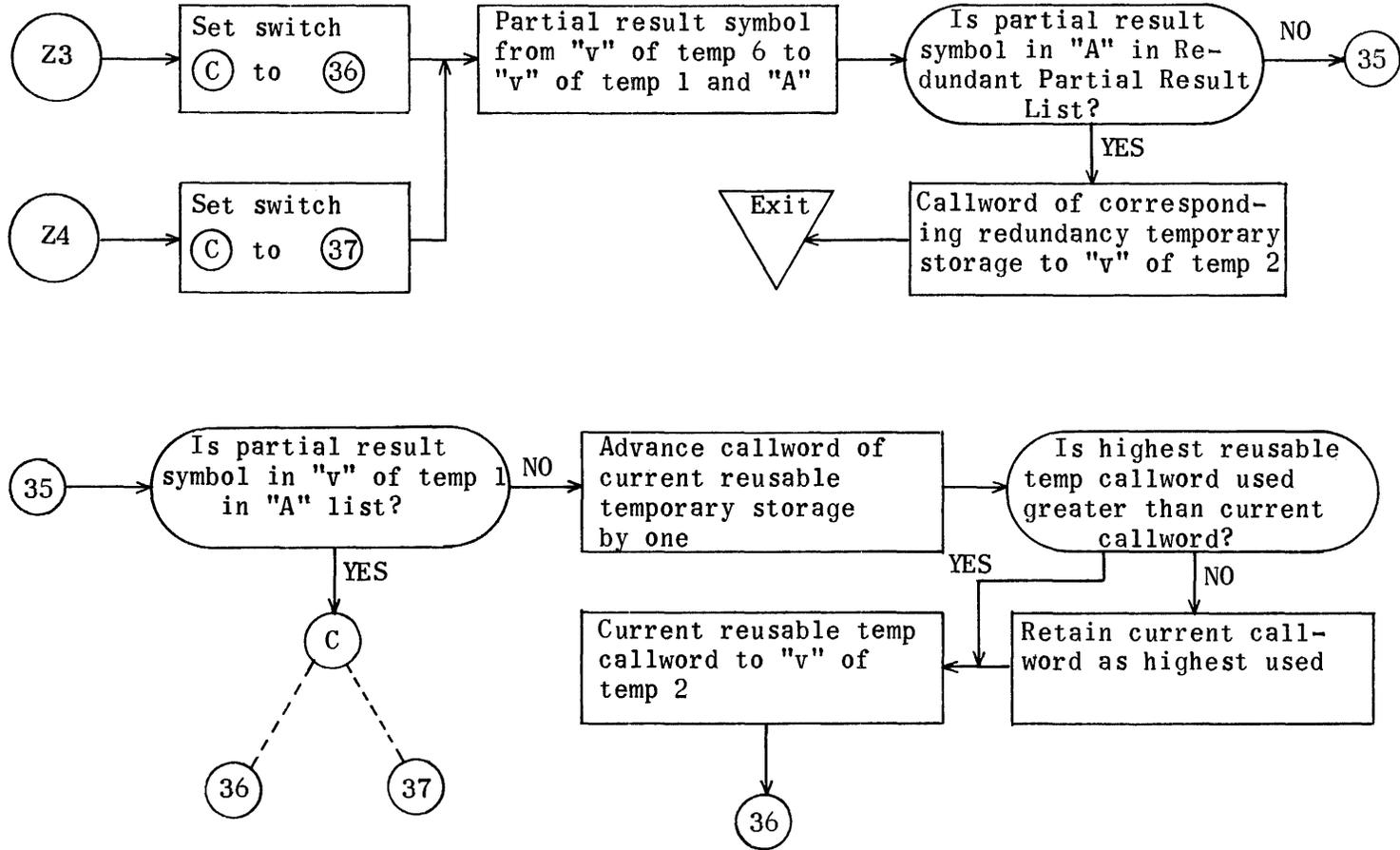
Equation Generation Subroutine to Obtain Pertinent Temporary Storage Callword (TR)



Equation Generation Subroutines for Fixed Point Operators



1401



REGIONS FOR EQUATION GENERATION NO. 3

RE UP421	Uniprint	
RE EP537	Machine Error Routine	
RE BQ632		}
RE WA653	Routine to Print Error Heading	
RE OP1047	Op. Control Routine	
RE CW1211	Constant Callword Routine	
RE BG2512		
RE GE2542		
RE EG2603		
RE GY2730		
RE GZ3030		
RE NZ3120		
RE ZZ3150		
RE GF3176		
RE GG3262		
RE GH3357		
RE GI3457		
RE GJ3564		
RE GK3624		
RE GL3661		
RE GM3713		
RE GN3764		
RE GP4043		
RE GQ4131		
RE GR4210		
RE GW4277		
RE GX4376		
RE GA4472		
RE GB4545		
RE GS4600		
RE GT4716		
RE GU5014		
RE GV5073		
RE SI5133		
RE TR5154		
RE GC5171		
RE TI5245		
RE T05316		
RE LG5322		
RE TT5324	Temporary Storage	
RE RB5360	"Generated Routine" Buffer (170 ₈ words)	
RE RA5550	Relative Address List	}
RE XQ5561	"Q" List	
RE XA5761	"A" List	
RE RL6161	Redundant Partial Result List	
RE EL6261	Expanded List	
RE FL7161	Op. File 1 Item for Generated Routine	
RE CI7361	"Generated Relative Constants" Image	
		Inputs from equation Redundancy check

REGIONS FOR EQUATION GENERATION NO. 3 (continued)

RE DL40102	Dimension List
RE RI65000	"Generated Routine" Image (used when Routine exceeds routine buffer, RB)
RE II5245	Dummy Instructions

Equation Generation No. 3

0	IA	BG		Begin Generation
	MJ	0	[30000]	Exit from phase
1	TP	GC50	RA4	Preset initial relative constant callword (10000 ₈)
2	TP	GC51	RA5	Preset initial redundancy temp callword less 1
3	TP	GC52	RA6	Preset initial reusable temp callword less 1
4	TP	GC52	RA7	Set highest temp CW = initial reus. temp CW less 1
5	TP	EL	RB	EQ. CW & # lines subj. add. mod. → 1st line of prelude
6	TP	GC17	RB3	#inputs = Zero → 4th line of prelude
7	TP	GC17	RB4	#outputs = Zero → 5th line of prelude
10	TP	EL1	RB5	Sentence number → 6th line of prelude
11	TP	II50	R36	Exit line → 1st line of Generated routine
12	TP	GC20	SI1	Preset add. in routine buffer for 2nd line Gen. routine
13	TP	GC27	SI6	Preset initial address in routine image
14	TP	GC30	SI12	Preset initial address in relative constant image
15	TV	GC21	GG45	Preset switch (A)
16	TP	GC17	TT	Zeroize temp 0
17	TP	GC17	TT1	Zeroize temp 1
20	TV	GC41	GI42	Preset switch (B)
21	TU	RA2	ZZ5	Preset repeat to search Redundant Partial Result List
22	TU	RA1	ZZ14	Preset repeat to search "A" List
23	TU	RA1	GZ24	Preset repeat to search "A" List
24	TU	RA2	TR2	Preset repeat to search Redundant Partial Result List
25	TU	RA1	NZ	Preset repeat to search "A" List
26	TU	RA2	NZ21	Preset repeat to search Redundant Partial Result List
27	MJ	0	GE	
	CA	BG30		

①	0	IA	GE		Generator Symbol Search			
		TU	RA10	GE2	Preset address of next word in Expanded List			
	1	RA	RA10	GC6	Advance address in Expanded List by one			
	2	SP	[30000]	0	Dummy instruction from Expanded List			
					→ A			
②	3	TP	A	TT5	Dummy instruction to temp 5			
	4	LT	6	Q	Get operator symbol from Op.code of dummy instruction			
	5	SP	Q	17	Operator symbol to "u" of A			
	6	AT	II16	A	Form <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>MJ</td><td>symbol</td><td>0</td></tr></table> → A	MJ	symbol	0
	MJ	symbol	0					
	7	RP	30027	GE40	Search list for operator symbol			
	10	TJ	GE11	GE11	Jump according to symbol			
	11	MJ	37	GY	Symbol for subscript manipulation			
	12	MJ	41	GF	Symbol for floating plus			
	13	MJ	52	GF2	Symbol for floating subtract			
14	MJ	53	GF4	Symbol for floating multiply				
③	15	MJ	54	GF6	Symbol for floating divide			
	16	MJ	55	GZ	Symbol for fixed plus			
	17	MJ	56	GZ3	Symbol for fixed subtract			
	20	MJ	57	GZ5	Symbol for fixed multiply			
	21	MJ	60	GZ16	Symbol for fixed divide			
	22	MJ	61	GL	Symbol for library operator			
	23	MJ	62	GN	Symbol for floating unary minus (neg)			
	24	MJ	63	GZ26	Symbol for fixed unary minus (neg)			
	25	MJ	64	GN2	Symbol for floating Abs. value			
	26	MJ	65	GZ30	Symbol for fixed Abs. value			
④	27	MJ	66	GP	Symbol for POW + 2			
	30	MJ	67	GP2	Symbol for POW - 2			
	31	MJ	70	GQ	Symbol for POW + 3			
	⑤	32	MJ	71	GQ2	Symbol for POW - 3		
		33	MJ	72	GX	Symbol for POW 1/2		
34	MJ	73	GX2	Symbol for POW - 1/2				
35	MJ	74	GW	Symbol for POW (4 to 63)				
36	MJ	75	GW2	Symbol for POW (-4 to -63)				
37	MJ	76	GR	Symbol for POW -1				
		CA	GE40					

	40	IA MJ CA	GE40 77 GE41	GA	Symbol for storage operator
⑥	0	IA RS	EG RA3	GC53	End Generation of Equation Number lines in object program body to A
	1	TJ	LG1	EG6	Is number of lines in object prog. body more than 1001g?
	2	RJ	WA	WA1	Yes; Type: SENTENCE____(EQUATION)
	3	TP	TO	UP3	Parameter for alarm text to type routine
	4	RJ	UP2	UP	Type: SENTENCE TOO LONG.
	5	MJ	0	BQ6	Rewind tapes and stop
	6	RS	RA4	GC50	Number of relative constants for object program to A
	7	TJ	GC44	EG11	Is number of relative constants more than 1000g?
	10	MJ	0	EG2	Yes, jump to type alarm
	11	RS	RA2	GC45	Number of redundancy temps for object program to A.
	12	TJ	GC44	EG14	Is number of redundancy temps more than 1000g?
	13	MJ	0	EG2	Yes, jump to type alarm
	14	TP	RA7	A	
	15	ST	GC52	RA6	Number of reusable temps for object program to A
	16	TJ	GC23	EG20	Is number of reusable temps more than 776g?
	17	MJ	0	EG2	Yes, jump to type alarm
⑦	20	SA	RA2	0	Number reusable temps + number redundancy temps
	21	SA	RA4	0	Add number of relative constants
	22	SA	RA3	0	Add number of lines in object program body
	23	TV	A	FL1	Number lines in object prog. including temps to Op. File 1
	24	SP	RA4	6	} Form codeword containing number of redundancy temps, number of reusable temps and number of relative constants for third line of prelude for routine.
	25	TV	A	RA6	
	26	TU	RA2	TT	
	27	SP	TT	11	
	30	SA	RA6	3	
	31	TP	A	RA5	Store codeword temporarily
	32	TP	II47	TT2	[MJ 0 1000] to temp 2 (jump to exit)
	33	RJ	SI	SI1	Store inst. in temp 2 in routine image
⑧	34	TP	SI6	A	
	35	EJ	GC27	EG41	Number of instructions in generated routine $\leq 170_8$?
	36	TP	SI12	A	No
	37	ST CA	GC30 EG40	TT3	Number of relative constants in relative constant image to A

	IA	EG40			
40	MJ	0	EG45		
41	TP	SI12	A		
42	ST	GC30	TT3	Number of relative constants in relative constant image to A	
43	SA	SI1	0		
44	TJ	GC42	EG101	Number lines in Gen. routine including rel. const. $\leq 170_8$?	
⑨	45	TP	SI1	A	No
	46	ST	GC24	TT4	
	47	SA	GC31	17	
	50	TU	A	EG52	
	51	TV	SI6	EG53	
	52	RP	[30000]	EG54	Generated Instructions from routine buffer to current address in routine image on drum
	53	TP	RB	[30000]	
	54	TP	TT3	A	
	55	SA	GC31	17	
	56	TU	A	EG62	
	57	TV	TT4	TT1	
	60	TV	SI6	EG63	
	61	RA	EG63	TT1	
	62	RP	[30000]	EG64	Relative constants from relative constant image to routine image on drum following generated instructions
	63	TP	CI	[30000]	
	64	TV	EG63	EG53	
	65	RA	TT3	EG53	
	66	SS	GC27	0	
	67	TV	A	RI	Number lines in prelude & routine to 1st line of prelude
⑩	70	ST	GC40	RI1	Number lines subject to address modification to 2nd line of prelude
	71	TP	RA5	RI2	Codeword to third line of prelude for routine
	72	TP	EL	A	Sentence callword from first word in Expanded List to A
	73	TJ	GC46	EG75	Is callword for equation in pseudo operation? (22---)
	74	TJ	GC15	EG77	No, is callword for separate equation? (24--- or 25---)
⑪	75	RA	RA7	GC2	No, advance highest reusable temp callword by two
	76	TV	A	RI6	Callword to "v" of exit line for Generated Routine
	77	TP	GC16	OP1	Send parameter to write Generated Routine from drum
		CA	EG100		

		IA	EG100		
⑫	100	MJ	0	EG123	
	101	TP	TT3	A	Number of relative constants in rel. const. image to A
	102	SA	GC31	17	
	103	TU	A	EG105	
	104	TV	SI1	EG106	
	105	RP	[30000]	EG107	Relative constants from relative constant image to routine buffer in core following generated instructions
	106	TP	CI	[30000]	
	107	TV	EG106	SI1	
	110	RA	TT3	SI1	
	111	SS	GC24	0	
	112	TV	A	RB	Number lines in prelude and routine to 1st line of prelude
	113	ST	GC40	RB1	Number lines subject to address modification to 2nd line of prelude
	114	TP	RA5	RB2	Codeword to third line of prelude
⑬	115	TP	EL	A	Sentence callword from first word in Expanded List to A
	116	TJ	GC46	EG120	Is callword for equation in Pseudo operation? (22----)
	117	TJ	GC15	EG122	No, is callword for separate equation? (24---- or 25----)
⑭	120	RA	RA7	GC2	No, advance highest reusable temp callword by two
	121	TV	A	RB6	Callword to "v" of exit line for Generated Routine
	122	TP	GC43	OP1	Send parameter to write Generated Routine from core
⑮	123	RJ	OP	OP2	Write generated routine and Op. File 1 item on tape
	124	MJ	0	BG	Jump to exit from phase
		CA	EG125		

	IA	GY		Generate Subscript Instructions
①⑥ "Sub"	RJ	GS5	GS	Next word from Expanded List to temp 6
Operator	LQ	A	25	Indicator from Op. code of word to "u"
				of A
	2	AT	II16	A
	3	RP	30007	GY14
	4	TJ	GY5	GY5
	5	MJ	0	GY15
	6	MJ	1	GY20
	7	MJ	2	GY22
	10	MJ	3	GY30
	11	MJ	4	GY34
	12	MJ	5	GY42
	13	MJ	6	GY47
	14	MJ	7	GY61
①⑦ Ind	15	RA	RA3	GC1
= 0				Adv. current rel. address by 3 in "u"
				and "v"
	16	RJ	GS12	GV31
	17	MJ	0	GY73
①⑧ Ind	20	RA	RA3	GC2
= 1				Adv. current rel. address by 2 in "u"
				and "v"
	21	MJ	0	GY73
①⑨ Ind	22	RA	RA3	GC
= 2				Adv. current rel. address by 4 in "u"
				and "v"
	23	RA	RA10	GC6
				Adv. add. in Exp. List by 1 → Add. of
				P.R. value
	24	TU	A	GV2
				Add. of word following last subs. in
				Exp. List → "u" of TP
	25	RJ	GV21	GV11
	26	RJ	GS12	GV7
	27	MJ	0	GY56
②⑩ Ind	30	RA	RA3	GC1
= 3				Adv. current rel. address by 3 in "u"
				and "v"
	31	TU	RA10	GV2
				Add. of last S.S. in Exp. List → "u"
				of TP
	32	RA	RA10	GC6
				Adv. Add. in Exp. List by 1 → add. of
				P.R. value
	33	MJ	0	GY67
②⑪ Ind	34	RA	RA3	GC4
= 4				Adv. current rel. address by 5 in "u"
				and "v"
	35	RA	RA10	GC36
				Adv. Add. in Exp. List by 2 → Add. of
				P.R. value
	36	TU	A	GV2
				Add. of word following last S.S. in
				Exp. List → "u" of TP
	37	RJ	GV21	GV11
		CA	GY40	
				To R4

		IA	GY40		Generate Subscript Ins. (cont.)
	40	RJ	GS12	GV7	To R2
	41	MJ	0	GY55	
②② Ind = 5	42	RA	RA3	GC	Adv. current rel. address by 4 in "u" and "v"
	43	RA	RA10	GC36	Adv. add. in Exp. List by 2 → Add. of P.R. value
	44	SS	GC6	0	Dec. "u" of A by one → Add. last subs. in Exp. List.
	45	TU	A	GV2	Add. of last S.S. in Exp. List → "u" of TP
	46	MJ	0	GY66	
②③ Ind = 6	47	RA	RA3	GC5	Adv. current rel. address by 6 in "u" and "v"
	50	RA	RA10	GC37	Adv. add. in Exp. List by 3 → Add. of P.R. value
	51	TU	A	GV2	Add. of word following last S.S. in Exp. List → "u" of TP
	52	RJ	GV21	GV11	To R4
	53	RJ	GS12	GV7	To R2
	54	RJ	GS12	GV	To R1
	55	RJ	GS12	GV	To R1
②④	56	TP	TT3	TT2	Generated [SA — 0] inst → temp 2
	57	RJ	SI	SI1	Store inst. in temp 2 in routine image
	60	MJ	0	GY70	
②⑤ Ind = 7	61	RA	RA3	GC4	Adv. current rel. add. by 5 in "u" and "v"
	62	RA	RA10	GC37	Adv. add. in Exp. List by 3 → add. of P.R. value
	63	SS	GC6	0	Dec. "u" of A by one → add. of last S.S. in Exp. list
	64	TU	A	GV2	Add. of last S.S. in Exp. List → "u" of TP
	65	RJ	GS12	GV	To R1
	66	RJ	GS12	GV	To R1
	67	RJ	GS12	GV	To R1
②⑥	70	TU	RA10	GY71	Add of P.R. value in Exp. List → "u" of NI
	71	TP	[30000]	TT6	P.R. value → "v" of temp 6
	72	RA	RA10	GC6	Adv. add. in Exp. List by 1 in "u"
②⑦	73	RJ	GS12	GV22	To R5
	74	RJ	GS12	GV27	To R6
②⑧	75	TP	II42	TT2	[TP A 30000] → Temp 2
	76	RJ	ZZ13	ZZ2	
	77	MJ	0	ZZ22	
		CA	GY100		

		IA	GZ	
②9	Fixed	TP	II12	TT3
	add.	RJ	GZ53	GZ40
		MJ	0	ZZ22
③0	Fixed	TP	II13	TT3
	subt.	MJ	0	GZ1
③1	Fixed	RJ	GS5	GS
	mult.	TP	II24	TT2
		ZJ	GZ12	GZ10
		10	RJ	GT27
				GT21
		11	MJ	0
				GZ12
		12	RJ	GT13
				GT5
		13	RA	RA3
				GC3
		14	RJ	SI
				SI1
		15	MJ	0
				GY75
③2	Fixed	TP	II37	TT3
	div.	RJ	GZ53	GZ40
		RA	RA3	GC3
		21	RJ	SI
				SI1
		22	TP	II41
				TT2
		23	TP	TT1
				A
		24	RP	[30000]
				GE
		25	EJ	XA
				ZZ22
③3	Fixed	TP	II31	TT2
	Unary	27	MJ	0
	minus			GZ31
	Fixed	30	TP	II44
	abs.			TT2
		31	RJ	GS5
				GS
		32	ZJ	GZ34
				GZ33
		33	RJ	GT27
				GT21
		34	TV	TT6
				TT1
		35	TP	TT1
				A
		36	MJ	0
				NZ
			CA	GZ37

Generate Fixed Point Inst.

[At 30000 A] → Temp 3
To Z1

[ST 30000 A] → Temp 3

Next word from Expanded List to temp 6

[MP A 30000] → temp 2

Check indicator from op. code of
word in temp 6

Indicator = 0 (to S33)

Indicator = 1 (to S29)

Adv. current relative address by one

Store inst. in temp 2 in routine image

[DV 30000 A] → temp 3

To Z1

Adv. current rel. address by 1 in
"u" and "v"

Store inst. in temp 2 in routine image

[TP 0 A] → temp 2

Partial result symbol → "A" register

Is partial result symbol in "A" List?

Yes → ZZ22; No → GE

[TN A A] → temp 2

[TM A A] → temp 2

Next word from Expanded List to temp 6

Check Indicator from op. code of
word in temp 6

Indicator = 0. (to S33)

Ind. = 1; P.R. symbol → "v" of temp 1

P.R. → "v" of A

		IA	GZ40		
ⓩ1	40	RJ	GS5	GS	Next word from Expanded List to temp 6
	41	ZJ	GZ54	GZ42	Check indicator in op. code of word (to GZ42 if ind = 0)
Ind =	42	RA	RA3	GC3	Adv. current relative address by one
0	43	SP	TT5	17	
	44	TU	A	TT	Operand symbol from "v" of temp 5 to "u" of temp 0
	45	RJ	GT27	GT22	To S34
	46	TU	A	TT3	Operand or temp callword to "u" of temp 3
ⓩ8	47	TP	II1	TT2	[TP 30000 A] → to temp 2
	50	RJ	GS12	GI17	To S32A
	51	TP	TT3	TT2	Instruction from temp 3 to temp 2
	52	RJ	ZZ13	ZZ	
	53	MJ	0	[30000]	
ⓩ9	54	TJ	GC14	GZ64	To GZ64 if indicator = 1
Ind =	55	TP	II31	TT2	Ind = 2; [TN A A] to temp 2
2	56	RJ	SI	SI1	Store instruction in temp 2 in routine image
	57	TU	TT5	TT	Operand symbol from "u" of temp 5 to "u" of temp 0
	60	RA	RA3	GC3	Advance current relative address by one
	61	TP	II12	TT2	[AT 30000 A] → temp 2
	62	RJ	GT27	GT22	To S34
	63	MJ	0	GZ52	
Ind =	64	SP	TT5	17	Ind = 1
1	65	TU	A	TT	Operand symbol from "v" of temp 5 to "u" of temp 0
	66	TP	TT3	TT2	Instruction from temp 3 to temp 2
	67	MJ	0	GZ62	
		CA	GZ70		

34A

0	IA	NZ		
	RP	[30000]	NZ11	Is partial result symbol in "A" List?
1	EJ	XA	NZ2	Yes → NZ2; no → NZ11
2	RA	RA3	GC3	Advance current relative address by one
3	RJ	SI	SI1	Store instruction in temp 2 in routine image
4	TP	II31	TT4	Set register indicator to "A" in "u" and "v"
5	TP	II42	TT2	[TP A 30000] → temp 2
6	TP	TT1	A	Partial result symbol → "v" of A
7	RJ	NZ21	NZ21	To Z2 (search Redundant P.R. List for P.R.)
10	MJ	0	GE	Exit - P.R. in "A" List and not in Redundant P.R. List

34B

11	RJ	NZ21	NZ21	To Z2 (search Red. P.R. List when P.R. not in "A" List)
12	RA	RA6	GC3	Advance current reusable temp callword by one
13	TJ	RA7	NZ15	Is highest temp callword used > current callword?
14	TP	A	RA7	No; retain current temp callword as highest used
15	TV	A	TT2	Reusable temp callword → "v" of temp 2
16	RA	RA3	GC3	Advance current relative address by one
17	RJ	SI	SI1	Store instruction in temp 2 in routine image

Z2

20	MJ	0	GE	
21	RP	[30000]	[30000]	Is partial result symbol in Redundant P.R. List?
22	EJ	RL	NZ23	Yes → NZ23; No → repeat exit
23	TP	RA2	A	jn → "u" & "v" of A
24	SS	Q	0	jn - (jn - r) → "v" of A
25	SA	RA5	0	Base redundancy temp callword + r → "v" of A
26	TV	A	TT2	Redundancy temp callword to "v" of temp 2
27	MJ	0	NZ16	
	CA	NZ30		

		IA	ZZ		
③	0	TV	GZ25	ZZ15	
	1	MJ	0	ZZ3	
④	2	TV	GC22	ZZ15	
	3	TV	TT6	TT1	Partial result symbol to "v" of temp 1
	4	TP	TT1	A	P.R. symbol → "v" of A
	5	RP	[30000]	ZZ14	Is P.R. in Redundant P.R. List?
	6	EJ	RL	ZZ7	Yes to ZZ7 ; no to ZZ14
	7	TP	RA2	A	jn → "u" and "v" of A
	10	SS	Q	0	jn - (jn - r) → "v" of A
	11	SA	RA5	0	Base redundancy temp. callword + r → "v" of A
	12	TV	A	TT2	Redundancy temp. callword → "v" of temp 2
	13	MJ	0	[30000]	Exit
⑤	14	RP	[30000]	ZZ16	Is partial result symbol in "A" List?
	15	EJ	XA	[30000]	No to ZZ16
	16	RA	RA6	GC3	Adv. current reusable temp callword by one
	17	TJ	RA7	ZZ21	Is highest temp used > current temp?
	20	TP	A	RA7	No, retain current temp callword as highest used
	21	TV	A	TT2	Current temp. callword → "v" of temp 2
⑥	22	RA	RA3	GC3	Advance current relative address by one
	23	RJ	SI	SI1	Store inst. in temp. 2 in routine image
⑦	24	TP	II31	TT4	Set register indicator to "A" in "u" and "v"
	25	MJ	0	GE	
		CA	ZZ26		

		IA	GF	
④⑩ Fl.	0	TP	II20	TT3
plus	1	MJ	0	GF7
④⑪ Fl.	2	TP	II21	TT3
subt.	3	MJ	0	GF7
④⑫ Fl.	4	TP	II22	TT3
mult.	5	MJ	0	GF7
④⑬ Fl.	6	TP	II23	TT3
divide	7	RJ	GS5	GS
④⑭	10	LQ	A	25
	11	AT	II16	A
	12	RP	30047	GF63
	13	TJ	GF14	GF14
	14	MJ	0	GG
	15	MJ	1	GG6
④⑮	16	MJ	2	GG10
	17	MJ	3	GG15
	20	MJ	4	GG17
	21	MJ	5	GG41
	22	MJ	6	GG55
	23	MJ	7	GG46
	24	MJ	10	GG50
	25	MJ	11	GH
	26	MJ	12	GH5
④⑯	27	MJ	13	GH10
	30	MJ	14	GH14
	31	MJ	15	GH21
	32	MJ	16	GH24
	33	MJ	17	GH40
	34	MJ	20	GH45
	35	MJ	21	GH50
	36	MJ	22	GH56
	37	MJ	23	GH66
		CA	GF40	

Generate Floating Point Inst.
 [FA 30000 30000] → temp 3

[FS 30000 30000] → temp 3

[FM 30000 30000] → temp 3

[FD 30000 30000] → temp 3

Next word from Expanded List to temp 6

Indicator from op. code of word to
 "u" of A

MJ	INDICATOR	00000	→ A
----	-----------	-------	-----

Search list for indicator

Jump according to indicator

Ind. = 0

Ind. = 1

Ind. = 2

Ind. = 3

Ind. = 4

Ind. = 5

Ind. = 6

Ind. = 7

Ind. = 10

Ind. = 11

Ind. = 12

Ind. = 13

Ind. = 14

Ind. = 15

Ind. = 16

Ind. = 17

Ind. = 20

Ind. = 21

Ind. = 22

Ind. = 23

(47)

	IA	GF40			
40	MJ	24	GH72	Ind. = 24	
41	MJ	25	GJ	Ind. = 25	
42	MJ	26	GJ6	Ind. = 26	
43	MJ	27	GJ11	Ind. = 27	
44	MJ	30	GJ20	Ind. = 30	
45	MJ	31	GJ27	Ind. = 31	
46	MJ	32	GJ32	Ind. = 32	
47	MJ	33	GI	Ind. = 33	
50	MJ	34	GI7	Ind. = 34	
(48)	51	MJ	35	GI13	Ind. = 35
	52	MJ	36	GI21	Ind. = 36
	53	MJ	40	GI25	Ind. = 40
	54	MJ	42	GI33	Ind. = 42
	55	MJ	43	GI36	Ind. = 43
	56	MJ	44	GI43	Ind. = 44
	57	MJ	45	GI50	Ind. = 45
	60	MJ	46	GI54	Ind. = 46
	61	MJ	47	GI61	Ind. = 47
	62	MJ	50	GI65	Ind. = 50
	63	MJ	51	GI71	Ind. = 51
	CA	GF64			

		IA	GG		Generate Floating Point (cont.)
④9	Ind	0	RJ	GS12	GS42
= 0		1	RA	RA3	GC
		2	RJ	GS12	GS6
		3	RJ	GS12	GS27
		4	RJ	GS12	GT2
		5	MJ	0	GK
④50	Ind	6	RJ	GS12	GS45
= 1		7	MJ	0	GG1
④51	Ind	10	RJ	GS12	GS42
= 2		11	RA	RA3	GC1
		12	RJ	GS12	GS32
		13	RJ	GS12	GT60
		14	MJ	0	GK
④52	Ind	15	RJ	GS12	GS45
= 3		16	MJ	0	GG11
④53	Ind	17	RA	RA3	GC
= 4		20	TP	II21	A
		21	EJ	TT3	GG27
		22	RJ	GS12	GS42
		23	RJ	GS12	GS6
		24	RJ	GS12	GS27
		25	RJ	GS12	GT70
		26	MJ	0	GK
④54		27	RJ	GS12	GS60
		30	TP	II5	TT2
		31	RJ	GS12	GS7
		32	TP	II6	TT2
		33	LQ	TT6	25
		34	TU	RA3	TT2
		35	RJ	GS12	GV4
④55		36	TP	II40	TT2
		37	RJ	SI	SI1
			CA	GG40	

To S11
 Adv. current rel. add. by 4 in "u"
 and "v"
 To S2
 To S6
 To S28
 To S12
 To S11
 Adv. current rel. add. by 3 in "u"
 and "v"
 To S7
 To S44
 To S12
 Adv. current rel. add. by 4 in "u"
 and "v"
 [FS 30000 30000] → A
 Is floating subtract inst. in temp 3?
 No; to S11
 To S2
 To S6
 To S46
 To S16
 [TV A 30000] → temp 2
 To S2A
 [RA 30000 30000] → temp 2
 Current relative address to "u" of
 temp 2
 To R3
 [FA Q 30000] → temp 2
 Store inst. in temp 2 in routine image

		IA	GG40		
		MJ	0	GK	
(56)	Ind	41	RA	RA3	GC1
= 5					
		42	RJ	GS12	GS62
		43	RJ	GS12	GS32
		44	RJ	GS12	GT64
(57)		45	MJ	0	[30000]
(58)	Ind	46	RA	RA3	GC2
= 7					
		47	MJ	0	GG43
(59)	Ind	50	RA	RA3	GC1
= 10					
		51	RJ	GS12	GS62
		52	RJ	GS12	GS32
		53	RJ	GS12	GT52
		54	MJ	0	GG45
(60)	Ind	55	TP	II21	A
= 6		56	EJ	TT3	GG64
		57	RA	RA3	GC1
		60	RJ	GS12	GS42
		61	RJ	GS12	GS32
		62	RJ	GS12	GT50
		63	MJ	0	GK
(61)		64	RA	RA3	GC
		65	RJ	GS12	GS60
		66	RJ	GS12	GS70
		67	RJ	GS12	GS32
		70	TP	II40	TT3
		71	LQ	TT5	25
		72	TV	Q	TT3
		73	RJ	GS12	GT43
		74	MJ	0	GK
			CA	GG75	

Adv. current rel. address by 3 in "u" and "v"

To S17

To S7

To S45

Switch (A)

Adv. current rel. address by 2 in "u" and "v"

Adv. current rel. address by 3 in "u" and "v"

To S17

To S7

To S42

To switch (A)

[FS 30000 30000] → A

Is floating subt. inst. in temp 3?

No, advance current relative address by three

To S11

To S7

To S41

Adv. current rel. address by 4 in "u" and "v"

To S16

To S20

To S7

[FA Q 30000] → temp 3

Operand symbol from "u" of temp 5 to "v" of temp 3

To S39A

				Generate Floating Point (cont.)	
⑥2	Ind	0	IA RA	GH RA3 GC1	Adv. current rel. address by 3 in "u" and "v"
= 11		1	RJ	GS12 GS13	To S3
		2	RJ	GS12 GT73	To S47
		3	RJ	GS12 GT16	To S32
		4	MJ	0 GG45	To switch(A)
⑥3	Ind	5	RA	RA3 GC3	Adv. current rel. address by 1 in "u" and "v"
= 12		6	RJ	GS12 GT54	To S43
		7	MJ	0 GK3	
⑥4	Ind	10	RA	RA3 GC1	Adv. current rel. address by 3 in "u" and "v"
= 13		11	RJ	GS12 GS13	To S3
		12	RJ	GS12 GS27	To S6
		13	MJ	0 GH3	
⑥5	Ind	14	RA	RA3 GC1	Adv. current rel. address by 3 in "u" and "v"
= 14		15	RJ	GS12 GS13	To S3
		16	RJ	GS12 GT73	To S47
		17	RJ	GS12 GT30	To S35
		20	MJ	0 GG45	To switch(A)
⑥6	Ind	21	RA	RA3 GC3	Adv. current rel. address by 1 in "u" and "v"
= 15		22	RJ	GS12 GT33	To S36
		23	MJ	0 GK3	
⑥7	Ind	24	TP	II21 A	[FS 30000 30000] → A
= 16		25	EJ	TT3 GH31	Is floating subt. inst. in temp 3?
		26	RA	RA3 GC3	No, advance current relative address by one
		27	RJ	GS12 GT36	To S37
⑥8		30	MJ	0 GK3	
		31	RA	RA3 GC2	Adv. current rel. address by 2 in "u" and "v"
		32	RJ	GS12 GS60	To S16
		33	TP	II40 TT2	[FA Q 30000] → temp 2
		34	LQ	TT5 25	Operand symbol from "u" of temp 5 to "v" of temp 5
		35	RJ	GT13 GT5	To S29
		36	RJ	SI SI1	Store inst in temp 2 in routine image
		37	MJ	0 GK3	
			CA	GH40	

		IA	GH40		Generate Floating Point (cont.)	
⑥9	Ind	40	RA	RA3	GC	Adv. current rel. address by 4 in "u" and "v"
=		17				
		41	RJ	GS12	GS42	To S11
		42	RJ	GS12	GS47	To S13
		43	RJ	GS12	GS32	To S7
		44	MJ	0	GH54	
⑦0	Ind	45	RA	RA3	GC	Adv. current rel. address by 4 in "u" and "v"
=		20				
		46	RJ	GS12	GS45	To S12
		47	MJ	0	GH42	
⑦1	Ind	50	RA	RA3	GC	Adv. current rel. address by 4 in "u" and "v"
=		21				
		51	RJ	GS12	GS35	To S9
		52	RJ	GS12	GS42	To S11
		53	RJ	GS12	GS40	To S10
		54	RJ	GS12	GT41	To S38
		55	MJ	0	GG45	To switch (A)
⑦2	Ind	56	RA	RA3	GC5	Adv. current rel. address by 6 in "u" and "v"
=		22				
		57	RJ	GS12	GS42	To S11
		60	RJ	GS12	GS47	To S13
		61	RJ	GS12	GS6	To S2
		62	RJ	GS12	GS13	To S3
		63	RJ	GS12	GS27	To S6
		64	RJ	GS12	GT14	To S31
		65	MJ	0	GG45	To switch (A)
⑦3	Ind	66	RA	RA3	GC5	Adv. current rel. address by 6 in "u" and "v"
=		23				
		67	RJ	GS12	GS45	To S12
		70	RJ	GS12	GS47	To S13
		71	MJ	0	GH61	
⑦4	Ind	72	RA	RA3	GC5	Adv. current rel. address by 6 in "u" and "v"
=		24				
		73	RJ	GS12	GS6	To S2
		74	RJ	GS12	GS53	To S14
		75	RJ	GS12	GS17	To S4
		76	RJ	GS12	GS42	To S11
		77	MJ	0	GH63	
			CA	GH100		

		IA	GI		Generate Floating Pt. (Function 61--- type)
(75) Ind	0	RA	RA3	GC4	Adv. current rel. add. by 5 in "u"
= 33					and "v"
	1	RJ	GS12	GS6	To S2
	2	RJ	GS12	GS42	To S11
	3	RJ	GS12	GS13	To S3
	4	RJ	GS12	GS27	To S6
	5	RJ	GS12	GT14	To S31
	6	MJ	0	GG45	To switch(A)
(76) Ind	7	RA	RA3	GC4	Adv. current rel. add. by 5 in "u"
= 34					and "v"
	10	RJ	GS12	GS6	To S2
	11	RJ	GS12	GS45	To S12
	12	MJ	0	GI3	
(77) Ind	13	RA	RA3	GC	Adv. current rel. add. by 4 in "u"
= 35					and "v"
	14	RJ	GS12	GS72	To S21
	15	RJ	GS12	GS42	To S11
	16	RJ	GS12	GS32	To S7
	17	RJ	GS12	GT46	To S40
	20	MJ	0	GG45	To switch(A)
(78) Ind	21	RA	RA3	GC	Adv. current rel. add. by 4 in "u"
= 36					and "v"
	22	RJ	GS12	GS72	To S21
	23	RJ	GS12	GS45	To S12
	24	MJ	0	GI16	
(79) Ind	25	RA	RA3	GC	Adv. current rel. add. 4 in "u" and "v"
= 40	26	RJ	GS12	GS23	To S8
	27	RJ	GS12	GS62	To S17
	30	RJ	GS12	GS32	To S7
	31	RJ	GS12	GT42	To S39
	32	MJ	0	GG45	To switch(A)
(80) Ind	33	RA	RA3	GC1	Adv. current rel. add. by 3 in "u"
= 42					and "v"
	34	RJ	GS12	GS23	To S8
	35	MJ	0	GI30	
(81) Ind	36	RA	RA3	GC1	Adv. current rel. add. by 3 in "u"
= 43					and "v"
	37	RJ	GS12	GS6	To S2
	CA	GI40			

					Gen. Fl. Pt. (Function 61--- Type) (cont.)
		IA	GI40		
	40	RJ	GS12	GS13	To S3
	41	RJ	GS12	GT14	To S31
(82)	42	MJ	0	[30000]	Switch(B)
(83) Ind	43	RA	RA3	GC	Adv. current rel. add. by 4 in "u" and "v"
= 44	44	RJ	GS12	GS6	To S2
	45	RJ	GS12	GS13	To S3
	46	RJ	GS12	GT73	To S47
	47	MJ	0	GH64	
(84) Ind	50	RA	RA3	GC2	Adv. current rel. add. by 2 in "u" & "v"
= 45	51	RJ	GS12	GS6	To S2
	52	RJ	GS12	GT2	To S28
	53	MJ	0	GK3	
(85) Ind	54	RA	RA3	GC	Adv. current rel. add. by 4 in "u" and "v"
= 46	55	RJ	GS12	GS6	To S2
	56	RJ	GS12	GS13	To S3
	57	RJ	GS12	GS27	To S6
	60	MJ	0	GH64	
(86) Ind	61	RA	RA3	GC2	Adv. current rel. add. by 2 in "u" and "v"
= 47	62	RJ	GS12	GS13	To S3
	63	RJ	GS12	GT16	To S35
	64	MJ	0	GI42	To switch(B)
(87) Ind	65	RA	RA3	GC2	Adv. current rel. add. by 2 in "u" and "v"
= 50	66	RJ	GS12	GS13	To S2
	67	RJ	GS12	GT30	To S19
	70	MJ	0	GI42	To switch(B)
(88) Ind	71	TP	II21	A	[FS 30000 30000] → A
= 51	72	EJ	TT3	GI77	Is floating subtract inst. in temp 3?
	73	RA	RA3	GC2	No, advance current relative address by two
	74	RJ	GS12	GS6	To S2
	75	RJ	GS12	GT70	To S46
	76	MJ	0	GK3	
(89)	77	RA	RA3	GC1	Adv. current rel. add. by 3 in "u" and "v"
		CA	GI100		
		IA	GI100		
	100	RJ	GS12	GS101	To S23
	101	RJ	GS12	GS60	To S16
	102	TP	II40	TT2	[FA Q 30000] → temp 2
	103	RJ	SI	SI1	Store instruction in temp 2 in routine image
	104	MJ	0	GK3	
		CA	GI105		

		IA	GJ		Generate Floating Point (cont.)
⑨⑩ Ind = 25	0	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	1	RJ	GS12	GS42	To S11
	2	RJ	GS12	GS47	To S13
	3	RJ	GS12	GS23	To S8
	4	RJ	GS12	GS32	To S7
⑨① Ind = 26	5	MJ	0	GJ16	
	6	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	7	RJ	GS12	GS45	To S12
⑨② Ind = 27	10	MJ	0	GJ2	
	11	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	12	RJ	GS12	GS32	To S7
	13	RJ	GS12	GS6	To S2
	14	RJ	GS12	GS42	To S11
	15	RJ	GS12	GS27	To S6
	16	RJ	GS12	GT42	To S39
	17	MJ	0	GG45	To switch (A)
⑨③ Ind = 30	20	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	21	RJ	GS12	GS42	To S11
	22	RJ	GS12	GS47	To S13
	23	RJ	GS12	GS72	To S21
	24	RJ	GS12	GS32	To S7
	25	RJ	GS12	GT46	To S40
⑨④ Ind = 31	26	MJ	0	GG45	To Switch (A)
	27	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	30	RJ	GS12	GS45	To S12
⑨⑤ Ind = 32	31	MJ	0	GJ22	
	32	RA	RA3	GC4	Adv. current rel. address by 5 in "u" and "v"
	33	RJ	GS12	GS72	To S21
	34	RJ	GS12	GS35	To S9
	35	RJ	GS12	GS42	To S11
	36	RJ	GS12	GS40	To S10
	37	MJ	0	GJ25	
		CA	GJ40		

		IA	GK		Generate Floating Point (cont.)
(96)	0	TU	RA10	GK2	Address of next word in Expanded List → "u" of TV
	1	RA	RA10	GC6	Advance address in Expanded List by 1 in "u"
	2	TV	[30000]	TT6	Partial result symbol from Expanded List to "v" of temp 6
(97)	3	TP	II33	TT4	Set register indicator to "Q" in "u" and "v"
	4	TP	II41	TT3	[TP Q A] → temp 3
	5	RJ	GK34	GK13	To K1
	6	TP	TT3	A	Inst. in temp 3 → A
	7	EJ	II41	GK12	Is inst. in temp 3 = [TP Q A] ?
	10	RA	RA3	GC3	No; advance current relative address by one
	11	RJ	GS12	GT14	To S31 (store instruction in temp 3 in routine image)
	12	MJ	0	GE	
(K1)	13	TV	TT6	TT1	Partial result symbol from "v" of temp 6 to "v" of temp 1
	14	TU	RA2	GK16	Preset repeat to search Redundant P.R. List
	15	TP	TT1	A	Partial result symbol to "v" of A
	16	RP	[30000]	GK25	Is partial result symbol in Redundant P.R. List?
	17	EJ	RL	GK20	Yes, to GK20; no, to GK25
	20	TP	RA2	A	jn to "u" and "v" of A
	21	SS	Q	0	jn - (jn - r) to "v" of A
	22	SA	RA5	0	Base redundancy temp callword + r to "v" of A
	23	TV	A	TT3	Redundancy temp callword to "v" of temp 3
	24	MJ	0	GK34	
(98)	25	TU	RA	GK26	Preset repeat to search "Q" List
	26	RP	[30000]	GK30	Is partial result symbol in "Q" List?
	27	EJ	XQ	GK34	Yes, to GK34; no, to GK30
	30	RA	RA6	GC3	Advance current reusable temp callword by one
	31	TJ	RA7	GK33	Is highest temp callword used > current callword?
(99)	32	TP	A	RA7	No, retain current temp callword as highest used
(100)	33	TV	A	TT3	Reusable temp callword to "v" of temp 3
	34	MJ	0	[30000]	Exit
		CA	GK35		

101	Op.	0	IA	GL		Generate Library Routine Reference
			TP	GC7	Q	Mask for rightmost octal digit of "v"
						→ Q
		1	QT	TT5	TT7	Number of arguments for library routine
		2	TP	GC32	A	to temp 7
		3	AT	TT7	TT10	[10 0 3] → A
102	104	4	IJ	TT7	GL16	Set 10 line counter → 10 line for last
		5	RA	RA3	GC3	argument
		6	TP	II14	TT2	Have all arguments been generated?
		7	SP	TT5	17	Yes, advance current relative address
		10	TU	A	TT2	by one
		11	TV	TT5	TT2	[RJ _ _] → temp 2
		12	RJ	SI	SI1	Library routine callword from "v" of
		13	TP	II43	TT2	temp 5 to "u" of A
		14	RJ	SI	SI1	Library callword to "u" of RJ in temp 2
		15	MJ	0	GK	Library callword to "v" of RJ in temp 2
		16	RJ	GS5	GS	Store inst. in temp 2 in routine image
		17	LQ	A	25	[10 0002 00000] to temp 2
		20	AT	II16	A	Store "10" line in temp 2 in routine
		21	RP	30006	GL31	image
		22	TJ	GL23	GL23	Next word from Expanded List to temp 6
		23	MJ	0	GM	Indicator from op. code of word to
103		24	MJ	1	GM12	"u" of A
		25	MJ	2	GM17	[MJ indicator 00000] to "A" register
		26	MJ	3	GM25	Search list for indicator
		27	MJ	4	GM30	Jump according to indicator
		30	MJ	5	GM43	Ind.= 0
		31	MJ	33	GM46	Ind.= 1
		CA	GL32			Ind.= 2
						Ind.= 3
						Ind.= 4
					Ind.= 5	
					Ind.= 33	

				Generate Library Routine Ref. (cont.)	
⑩⑤ = 0	Ind	0	IA RA	GM RA3 GC3	Adv. current rel. add. by 1 in "u" and "v"
		1	TP	II1 TT2	[TP 30000 A] → temp 2
		2	TV	TT5 TT2	Library routine callword to "v" of TP inst. in temp 2
		3	TU	TT6 TT	Argument callword from "u" of temp 6 to "u" of temp 1
⑩⑥		4	RJ	GT27 GT22	To S34
		5	RJ	SI SI1	Store inst. in temp 2 in routine image
		6	TP	TT10 TT2	"10" line for argument to temp 2
		7	RJ	SI SI1	Store "10 line" in temp 2 in routine image
⑩⑦ = 1	Ind	10	RS	TT10 GC11	Decrease "10" line counter by one
		11	MJ	0 GL4	
		12	RA	RA3 GC3	Adv. current rel. add. by 1 in "u" and "v"
		13	TP	II1 TT2	[TP 30000 A] → temp 2
⑩⑧ = 2	Ind	14	TU	TT4 TT2	"u" of register indicator to "u" of TP inst. in temp 2
		15	TV	TT5 TT2	Library routine callword to "v" of TP inst. in temp 2
		16	MJ	0 GM5	
		17	RA	RA3 GC	Adv. current rel. add. by 4 in "u" and "v"
⑩⑨ = 3	Ind	20	RJ	GS12 GS56	To S15
		21	RJ	GS12 GS106	To S2
		22	RJ	GS12 GS27	To S6
		23	TP	II1 TT2	[TP 30000 A] → temp 2
⑩⑩ = 4	Ind	24	MJ	0 GM15	
		25	RA	RA3 GC	Adv. current rel. address by 4 in "u" and "v"
		26	RJ	GS12 GS45	To S12
		27	MJ	0 GM21	
⑩⑪	Ind	30	RA	RA3 GC1	Adv. current rel. address by 3 in "u" and "v"
		31	RJ	GS12 GS56	To S15
		32	RJ	GS12 GS32	To S7
		33	TP	II1 TT3	[TP 30000 A] → temp 3
⑩⑫	Ind	34	TU	TT6 TT3	Argument callword to "u" of TP inst. in temp 3
		35	TV	TT5 TT3	Library routine callword to "v" of TP inst. in temp 3
		36	RJ	GS12 GT43	To S39A (inst. in temp 3 to relative constant image)
		37	TP CA	TT10 GM40	"10" line for argument to temp 3

	40	IA RJ	GM40 SI	SI12	"10" line in temp 3 to relative constant image
	41	RS	TT10	GC11	Decrease "10" line counter by one
	42	MJ	0	GL4	
①112 = 5	Ind 43	RA	RA3	GC1	Adv. current rel. add. by 3 in "u" and "v"
	44	RJ	GS12	GS45	To S12
	45	MJ	0	GM32	
①113 = 33	Ind 46	RA	RA3	GC2	Adv. current rel. add. by 2 in "u" and "v"
	47	RJ	GS12	GS106	To S25
	50	MJ CA	0 GM51	GM23	

		IA	GN	
(114)	F1.	0	TP	II3
	Unary	1	MJ	0
	minus	2	TP	II2
(115)	F1.	3	RJ	GS5
	Abs. Value	4	LQ	A
(116)				
		5	AT	II16
		6	RP	30006
		7	TJ	GN10
		10	MJ	0
(117)		11	MJ	1
		12	MJ	2
		13	MJ	3
		14	MJ	12
		15	MJ	15
		16	MJ	45
(118)	Ind	17	RJ	GS12
=0		20	RA	RA3
		21	RJ	GS12
		22	RJ	GS12
(119)		23	RJ	GN56
		24	RJ	GS12
		25	MJ	0
(120)	Ind	26	RJ	GS12
=1		27	MJ	0
(121)	Ind	30	RJ	GS12
=2		31	RA	RA3
		32	RJ	GS12
		33	TU	TT5
		34	RJ	GN56
		35	RJ	GS12
		36	MJ	0
(122)	Ind	37	RJ	GS12
=3			CA	GN40

Generate Floating Neg. and Abs. Value
 [TN 30000 Q] → temp 3

[TM 30000 Q] → temp 3
 Next word from Expanded List to temp 6
 Indicator from op. code of word to "u"
 of A

Form

MJ	indicator	00000
----	-----------	-------

 in A

Search list for indicator

Jump according to indicator

Ind. = 0

Ind. = 1

Ind. = 2

Ind. = 3

Ind. = 12

Ind. = 15

Ind. = 45

To S48

Adv. current rel. address by 4 in "u"
 and "v"

To S7

To S6

To NI

To S31 (store instruction in temp 3 in
 routine image)

To S12

To S48

Adv. current rel. address by 3 in "u"
 and "v"

To S7

Operand callword from "u" of temp 5
 to "u" of temp 3

To (N1)

To S39A (inst. from temp 3 to relative
 constant image)

To S12

				Gen. Fl. Neg. and Abs. Val. (cont.)
		IA	GN40	
		MJ	0	GN31
(123)	Ind	41	RJ	GT27
= 12		42	TU	TT2
				TT3
		43	RA	RA3
		44	MJ	0
Ind =		45	TU	TT4
15				TT3
		46	MJ	0
Ind =		47	RA	RA3
45		50	RJ	GS12
		51	MJ	0
(N1)		52	RJ	GK34
		53	TV	TT3
		54	SP	TT3
		55	TU	A
		56	MJ	0
			CA	GN57

Gen. Fl. Neg. and Abs. Val. (cont.)

To S33

Operand or temp callword to "u" of temp 3

Advance current relative address by one

"u" of register indicator to "u" of temp 3

Advance current relative address by two

To S2

To K1

Set "v" of register indicator

Set "u" of register indicator

		IA	GP		Generate Int. Power Inst.
(126)	POW	0	RJ	GP65	GP5
2		1	MJ	0	GK3
(127)	POW	2	RJ	GP65	GP5
-2		3	RJ	GS12	GU
		4	MJ	0	GK3
(P2)		5	RJ	GS5	GS
		6	LQ	A	25
		7	AT	II16	A
		10	RP	30005	GP17
		11	TJ	GP12	GP12
		12	MJ	0	GP20
		13	MJ	4	GP23
		14	MJ	5	GP35
		15	MJ	10	GP46
		16	MJ	11	GP57
		17	MJ	33	GP61
(128)	Ind	20	RA	RA3	GC3
=0		21	RJ	GS12	GU34
		22	MJ	0	GP65
(129)	Ind	23	RA	RA3	GC
=4		24	RJ	GS12	GV36
		25	RJ	GS12	GU54
		26	RJ	GS12	GS32
		27	TP	II22	TT3
		30	TU	TT5	TT3
		31	LQ	TT5	25
		32	TV	Q	TT3
		33	RJ	GS12	GT43
		34	MJ	0	GP65
(130)	Ind	35	RA	RA3	GC1
=5		36	RJ	GS12	GU24
		37	RJ	GS12	GS21
			CA	GP40	

To P2

To P2

To S49

Next word from Expanded List to temp 6
Indicator from op. code of word to "u"
of A
Form

MJ	indicator	00000
----	-----------	-------

 in "A"
Search list for indicator
Jump according to indicator
Ind. = 0
Ind. = 4
Ind. = 5
Ind. = 10
Ind. = 11
Ind. = 33
Adv. current rel. address by 1 in "u"
and "v"
To S56

Adv. current rel. address by 4 in "u"
and "v"
To S48
To S59
To S7
[FM 30000 30000] → temp 3
Operand callword to "u" of FM inst.
in temp 3

Same callword to "v" of FM inst. in
temp 3
To S39A (inst. from temp 3 to relative
constant image)

Advance current rel. address by 3 in
"u" and "v"
To S54
To S5

		IA	GP40		Generate Int. Power Inst. (cont.)
	40	RJ	GS12	GS64	To S18
	41	RA	RA3	GC3	Adv. current rel. address by 1 in "u" and "v"
	42	TP	II34	TT2	[FM A A] to temp 2
	43	RJ	SI	SI1	Store inst. in temp 2 in routine image
	44	RJ	GU33	GU27	To S55
	45	MJ	0	GP65	
(131) Ind	46	RJ	GS12	GS66	To S19
= 10	47	RA	RA3	GC	Adv. current rel. address by 4 in "u" and "v"
	50	RJ	GS12	GS76	To S22
	51	RJ	GS12	GS21	To S5
	52	RJ	GS12	GS64	To S18
	53	RA	RA3	GC3	Adv. current rel. address by 1 in "u" and "v"
	54	TP	II34	TT2	[FM A A] to temp 2
	55	RJ	SI	SI1	Store instruction in temp 2 in routine image
	56	MJ	0	GP65	
(132) Ind	57	RA	RA3	GC1	Adv. current rel. address by 3 in "u" and "v"
= 11	60	MJ	0	GP50	
(133) Ind	61	RA	RA3	GC1	Adv. current rel. address by 3 in "u" and "v"
= 33	62	RJ	GS12	GS6	To S2
	63	RJ	GS12	GS101	To S23
	64	RJ	GS12	GU52	To S58
	65	MJ	0	[30000]	Exit
		CA	GP66		

		IA	GQ		Generate Int. Power Inst. (cont.)	
(134)	POW	0	RJ	GQ54	GQ5	To P3
3		1	MJ	0	GK3	
(135)	POW	2	RJ	GQ54	GQ5	To P3
-3		3	RJ	GS12	GU	To S49
		4	MJ	0	GK3	
(P3)		5	RJ	GS5	GS	Next word from Expanded List to temp 6
		6	LQ	A	25	Indicator from op. code of word to "u"
		7	AT	II16	A	of A
		10	RP	30005	GQ17	Form MJ indicator 00000 in "A"
		11	TJ	GQ12	GQ12	Search list for indicator
		12	MJ	0	GQ20	Jump according to indicator
		13	MJ	4	GQ25	Ind.= 0
		14	MJ	5	GQ34	Ind.= 4
		15	MJ	10	GQ42	Ind.= 5
		16	MJ	11	GQ46	Ind.= 10
		17	MJ	33	GQ51	Ind.= 11
		20	RA	RA3	GC2	Ind.= 33
(136)	Ind	21	RJ	GS12	GU34	Adv. current rel. address by 2 in "u"
=0		22	TU	II33	TT2	and "v"
		23	RJ	SI	SI1	To S56
		24	MJ	0	GQ54	"Q" address to "u" of instruction in
(137)	Ind	25	RA	RA3	GC1	temp 2
=4		26	RJ	GS12	GV36	Store instruction in temp 2 in routine
		27	RJ	GS12	GS32	image
		30	RJ	GS12	GU14	Adv. current rel. address by 3 in "u"
		31	RA	RA3	GC2	and "v"
		32	RJ	GS12	GU10	To S48
		33	MJ	0	GQ55	To S7
(138)	Ind	34	RA	RA3	GC1	To S52
=5		35	RJ	GS12	GU24	Advance current rel. address by 2 in
		36	RJ	GU33	GU27	"u" and "v"
		37	RJ	GS12	GS21	To S51
			CA	GQ40		To S54
						To S55
						To S5

		IA	GQ40		Generate Int. Power Inst. (cont.)
	40	RJ	GS12	GU20	To S53
	41	MJ	0	GQ31	
Ind =	42	RA	RA3	GC	Adv. current rel. address by 4 in "u"
10					and "v"
	43	RJ	GS12	GS66	To S19
	44	RJ	GS12	GS76	To S22
	45	MJ	0	GQ37	
Ind =	46	RA	RA3	GC1	Adv. current rel. address by 3 in "u"
11					and "v"
	47	RJ	GS12	GS76	To S22
	50	MJ	0	GQ37	
Ind =	51	RA	RA3	GC2	Adv. current rel. address by 2 in "u"
33					and "v"
	52	RJ	GS12	GS6	To S2
	53	MJ	0	GQ40	
	54	MJ	0	[30000]	Exit
	55	RJ	GS12	GU4	To S50
	56	MJ	0	GQ54	
		CA	GQ57		

		IA	GR		Generate Int. Power Inst. (cont.)
(142)	POW	RJ	GS5	GS	Next word from Expanded List to temp 6
-1		LQ	A	25	Indicator from op. code of word to "u"
					of A
		2	AT	II16	A
		3	RP	30006	GR13
		4	TJ	GR5	GR5
		5	MJ	0	GR14
(143)		6	MJ	1	GR21
		7	MJ	4	GR25
		10	MJ	5	GR36
		11	MJ	10	GR50
		12	MJ	11	GR53
		13	MJ	33	GR57
(144)	Ind	14	TP	II23	TT2
=0		15	LQ	TT5	25
		16	RJ	GT13	GT5
		17	RA	RA3	GC3
		20	MJ	0	GR45
(145)	Ind	21	RA	RA3	GC3
=1		22	TP	II23	TT2
		23	TV	TT4	TT2
		24	MJ	0	GR45
(146)	Ind	25	RA	RA3	GC1
=4		26	RJ	GS12	GS66
		27	RJ	GS12	GS32
		30	TP	II23	TT3
		31	TU	TT6	TT3
		32	LQ	TT5	25
		33	TV	Q	TT3
		34	RJ	GS12	GT43
		35	MJ	0	GK3
(147)	Ind	36	RA	RA3	GC1
=5		37	TP	II32	TT2
			CA	GR40	

From MJ indicator 00000 to "A"
Search list for indicator
Jump according to indicator
Ind.= 0
Ind.= 1
Ind.= 4
Ind.= 5
Ind.= 10
Ind.= 11
Ind.= 33
[FD 30000 30000] → temp 2
Operand callword from "u" of temp 5 to
"v" of temp 5
To S29
Advance current relative address by one
Advance current relative address by one
[FD 30000 30000] → temp 2
"v" of register indicator → "v" of
temp 2
Advance current relative address by
three
To S19
To S7
[FD 30000 30000] → temp 3
Constant callword for floating point
one to "u" of temp 3
Operand callword to "v" of FD inst. in
temp 3
To S39A (inst. from temp 3 to relative
constant image)
Advance current relative address by
three
[SA 30000 0] → temp 2

		IA	GR40		Gen. Int. Power Inst. (cont.)
	40	TU	RA4	TT2	Relative constant callword to "u" of temp 2
	41	RJ	SI	SI1	Store inst. in temp 2 in routine image
	42	RJ	GS12	GS17	To S4
	43	RJ	GU33	GU27	To S55
(148)	44	TP	II23	TT2	[FD 30000 30000] → temp 2
(149)	45	TU	TT6	TT2	Constant callword for floating point one to "u" of temp 2
	46	RJ	SI	SI1	Store inst. in temp 2 in routine image
	47	MJ	0	GK3	
(150) Ind	50	RA	RA3	GC1	Adv. current rel. add. by 3 in "u" and "v"
=-10					To S23
	51	RJ	GS12	GS101	
	52	MJ	0	GR62	
(151) Ind	53	RA	RA3	GC1	Adv. current rel. add. by 3 in "u" and "v"
=-11					To S26
	54	RJ	GS12	GS114	To S4
	55	RJ	GS12	GS17	
	56	MJ	0	GR44	
(152) Ind	57	RA	RA3	GC2	Adv. current rel. address by 2 in "u" and "v"
=-33					To S23
	60	RJ	GS12	GS101	
	61	MJ	0	GR44	
	62	TP	II6	TT2	[RA 30000 30000] to temp 2
	63	RJ	GT13	GT5	To S29
	64	TU	RA3	TT2	Current relative address to "u" of temp 2
	65	RJ	SI	SI1	Store instruction in temp 2 in routine image
	66	MJ	0	GR44	
		CA	GR67		

	IA	GW		Generate Int. Power Inst. (cont.)			
(153) POW 0	RJ	GW64	GW10	To P4			
(4 to 63) 1	MJ	0	GK				
(154) POW 2	RJ	GW64	GW10	To P4			
(4 to 3	TU	RA10	GW5	Address of next word in Expanded			
-63)				List → "u" of TP			
	4	RA	RA10	Advance address in Expanded List			
				by 1 in "u"			
	5	TP	[30000] TT6	Next word in Expanded List to temp 6			
	6	RJ	GS12	To S49			
	7	MJ	0				
(P4)	10	RJ	GW73	Next word from Expanded List to temp 6			
	11	LQ	A	Indicator from op. code of next word			
				to "u" of A			
	12	AT	II16	Form <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>MJ</td><td>indicator</td><td>00000</td></tr></table> in "A"	MJ	indicator	00000
MJ	indicator	00000					
	13	RP	30005	Search list for indicator			
	14	TJ	GW15	Jump according to indicator			
	15	MJ	0	Ind. = 0			
	16	MJ	4	Ind. = 4			
	17	MJ	5	Ind. = 5			
	20	MJ	10	Ind. = 10			
	21	MJ	11	Ind. = 11			
	22	MJ	33	Ind. = 33			
(155) Ind	23	RA	RA3	Adv. current rel. address by 3			
= 0	24	RJ	GS12	To S56			
	25	RJ	GS12	To S57			
	26	TP	II35	[FM Q 30000] → temp 2			
	27	LQ	TT5				
	30	TV	Q	Operand callword from "u" of temp 5 to			
				"v" of temp 1			
	31	RJ	GT13	To S30			
	32	RJ	SI	Store inst. in temp 2 in routine image			
	33	MJ	0				
(156) Ind	34	RA	RA3	Advance current relative address by			
= 4				three			
	35	RJ	GS12	To S48			
	36	RJ	GS12	To S7			
	37	MJ	0				
		CA	GW40				

	IA	GW40		Generate Int. Power Inst. (cont.)	
	RA	RA3	GC1	Adv. current rel. address by 3 in "u" and "v"	
	RJ	GS12	GU20	To S53	
	RJ	GS12	GU10	To S51	
	RJ	GS12	GU42	To S57	
	RJ	GS12	GU4	To S50	
	MJ	0	GW64		
(157) =5	Ind	RA	RA3	GC1	Adv. current address by 3 in "u" and "v"
		RJ	GS12	GU24	To S54
		RJ	GU33	GU27	To S55
		RJ	GS12	GS21	To S5
		MJ	0	GW40	
(158) =10	Ind	RA	RA3	GC	Adv. current rel. add. by 4 in "u" and "v"
		RJ	GS12	GV31	To R7
		RJ	GS12	GS76	To S22
		MJ	0	GW51	
(159) =11	Ind	RA	RA3	GC1	Adv. current rel. address by 3 in "u" and "v"
		MJ	0	GW55	
(160) =33	Ind	RA	RA3	GC2	Adv. current rel. address by 2 in "u" and "v"
		RJ	GS12	GS6	To S2
		MJ	0	GW40	
		MJ	0	[30000]	Exit
(P5)		TU	RA10	GW67	Preset address of next word in Expanded List
		RA	RA10	GC6	Advance address in Expanded List by one
		TP	[30000]	TT6	Next word in Expanded List to temp 6
		TU	RA10	GW72	Preset address of next word in Expanded List
		TP	GC10	Q	
		QT	[30000]	A	Indicator from op. code of this word to op. code of A
		MJ	0	[30000]	
		RJ	GS12	GU14	To S52
		RA	RA3	GC1	Advance current relative address by three
		MJ	0	GW42	
		CA	GW77		

		IA	GX		Generate Int. Power Inst. (cont.)
(161) POW	0	RJ	GX67	GX5	To P1
1/2	1	MJ	0	GK3	
(162) POW	2	RJ	GX67	GX5	To P1
-1/2	3	RJ	GS12	GU	To S49
	4	MJ	0	GK3	
(P1)	5	TP	GC32	TT10	Preset "10" line counter to [10 00000 00003]
	6	RJ	GS5	GS	Next word from Expanded List to temp 6
	7	LQ	A	25	Indicator from op. code of this word to "u" of A
	10	AT	II16	A	Form [MJ indicator 00000] in A
	11	RP	30006	GX21	Search list for indicator
	12	TJ	GX13	GX13	Jump according to indicator
	13	MJ	0	GX22	Ind. = 0
	14	MJ	1	GX26	Ind. = 1
	15	MJ	4	GX32	Ind. = 4
(163)	16	MJ	5	GX44	Ind. = 5
	17	MJ	10	GX46	Ind. = 10
	20	MJ	11	GX47	Ind. = 11
	21	MJ	33	GX53	Ind. = 33
(165) Ind =0	22	RA	RA3	GC3	Adv. current rel. address by 1 in "u" and "v"
	23	TP	II30	TT2	[TP 30000 50051] → temp 2
	24	RJ	GT27	GT21	To S33
	25	MJ	0	GX56	
(166) Ind =1	26	RA	RA3	GC3	Advance current relative address by one
	27	TP	II30	TT2	[TP 30000 50051] → temp 2
	30	TU	TT4	TT2	"u" of register indicator to "u" of temp 2
	31	MJ	0	GX56	
(167) Ind =4	32	RA	RA3	GC1	Adv. current rel. add. by 3 in "u" and "v"
	33	RJ	GS12	GV36	To S48
	34	RJ	GS12	GS32	To S7
	35	TP	II30	TT3	[TP 30000 50051] → temp 3
	36	TU	TT5	TT3	Operand callword from "u" of temp 5 to "u" of temp 3
	37	RJ	GS12	GT43	To S39A (Inst. from temp 3 to relative constant image)
		CA	GX40		

	IA	GX40		
	40 RA	TT10	GC11	
	41 TP	A	TT3	
	42 RJ	SI	SI12	
	43 MJ	0	GX62	
(168) Ind	44 RJ	GS12	GU24	
=5	45 MJ	0	GX72	
(169) Ind	46 RJ	GS12	GX70	
=10	47 RJ	GS12	GS76	
(170) Ind	50 RA	RA3	GC1	
=11				
(171)	51 RJ	GS12	GS21	
	52 MJ	0	GX55	
Ind =	53 RA	RA3	GC2	
33				
	54 RJ	GS12	GS6	
(164)	55 TP	II30	TT2	
(172)	56 RJ	SI	SI1	
	57 RA	TT10	GC11	
	60 TP	A	TT2	
	61 RJ	SI	SI1	
(173)	62 RA	RA3	GC3	
	63 TP	II14	TT2	
	64 RJ	SI	SI1	
	65 TP	II43	TT2	
	66 RJ	SI	SI1	
	67 MJ	0	[30000]	
	70 RA	RA3	GC3	
	71 MJ	0	GV31	
	72 RJ	GU33	GU27	
	73 MJ	0	GX50	
	CA	GX74		

Gen. Int. Power (cont.)

Advance "10" line counter by one
 "10" line in "A" to temp 3
 "10" line in temps 3 to relative
 constant image

To S54

To S22

Advance current relative address by
 three
 To S5

Advance current relative address by
 two
 To S2

[TP 30000 50051] to temp 2
 Store inst. in temp 2 in routine image
 Advance "10" line counter by one in "v"
 "10" line in A to temp 2

Store inst. in temp 2 in routine image
 Advance current relative address by one
 [RJ 50051 50051] to temp 2

Store inst. in temp 2 in routine image
 [I0 00002 00000] to temp 2
 Store "10" line in temp 2 in routine
 image

Exit
 Advance current relative address by one
 To R7
 To S55

(174)	0	IA TP	GA II46	TT3	Generate Store (by or =) Instruction [TP 30000 30000] instruction to temp 3			
	1	TV	GC33	GG45	Set switch (A)			
	2	TV	GC33	GI42	Set switch (B)			
	3	RJ	GS5	GS	Next word from Expanded List to temp 6			
	4	LQ	A	25	Indicator from op. code of this word to "u" of A			
	5	AT	II16	A	Form <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>MJ</td><td>indicator</td><td>00000</td></tr></table> in A	MJ	indicator	00000
MJ	indicator	00000						
	6	RP	30042	GA52	Search list for indicator			
	7	TJ	GA10	GA10	Jump according to indicator			
	10	MJ	0	GB	Ind. = 0			
	11	MJ	1	GB10	Ind. = 1			
(175)	12	MJ	2	GB12	Ind. = 2			
	13	MJ	3	GB17	Ind. = 3			
	14	MJ	5	GG41	Ind. = 5			
	15	MJ	7	GG46	Ind. = 7			
	16	MJ	10	GG50	Ind. = 10			
	17	MJ	11	GH	Ind. = 11			
	20	MJ	12	GB27	Ind. = 12			
	21	MJ	13	GH10	Ind. = 13			
(176)	22	MJ	14	GH14	Ind. = 14			
	23	MJ	15	GB21	Ind. = 15			
	24	MJ	17	GH40	Ind. = 17			
	25	MJ	20	GH45	Ind. = 20			
	26	MJ	21	GH50	Ind. = 21			
	27	MJ	22	GH56	Ind. = 22			
	30	MJ	23	GH66	Ind. = 23			
	31	MJ	24	GH72	Ind. = 24			
(177)	32	MJ	25	GJ	Ind. = 25			
	33	MJ	26	GJ6	Ind. = 26			
	34	MJ	30	GJ20	Ind. = 30			
	35	MJ	31	GJ27	Ind. = 31			
	36	MJ	32	GJ32	Ind. = 32			
	37	MJ	33	GI	Ind. = 33			
		CA	GA40					

		IA	GA40		Gen. Store Inst. (cont.)
	40	MJ	34	GI7	Ind. = 34
	41	MJ	35	GI13	Ind. = 35
(178)	42	MJ	36	GI21	Ind. = 36
	43	MJ	40	GI25	Ind. = 40
	44	MJ	42	GI33	Ind. = 42
	45	MJ	43	GI36	Ind. = 43
	46	MJ	44	GI43	Ind. = 44
	47	MJ	45	GB24	Ind. = 45
	50	MJ	46	GI54	Ind. = 46
	51	MJ	47	GI61	Ind. = 47
	52	MJ	50	GI65	Ind. = 50
		CA	GA53		

		IA	GB		Gen. Store Inst. (cont.)	
①79 =0	Ind	0	RJ	GS12	GS42	To S11
		1	RA	RA3	GC	Advance current relative address by four
		2	RJ	GS12	GS6	To S2
		3	RJ	GS12	GS27	To S6
		4	TV	TT5	TT3	Callword of variable defined by equation to "v" of temp 3
		5	TP	TT3	TT2	Generated inst. to store result to temp 2
		6	RJ	SI	SI1	Store inst. in temp 2 in routine image
		7	MJ	0	EG	
①80 =1	Ind	10	RJ	GS12	GS45	To S12
		11	MJ	0	GB1	
①81 =2	Ind	12	RJ	GS12	GS42	To S11
		13	RA	RA3	GC1	Advance current relative address by three
		14	RJ	GS12	GS32	To S7
		15	RJ	GS12	GT41	To S38
		16	MJ	0	EG	
①82 =3	Ind	17	RJ	GS12	GS45	To S12
		20	MJ	0	GB13	
①83 =15	Ind	21	RA	RA3	GC3	Advance current relative address by one "A" or "Q" address from register indicator to "u" of temp 3
		22	TU	TT4	TT3	
		23	MJ	0	GB4	
①84 =45	Ind	24	RA	RA3	GC2	Advance current relative address by two
		25	RJ	GS12	GS6	To S2
		26	MJ	0	GB4	
①85 =12	Ind	27	RA	RA3	GC3	Advance current relative address by one
		30	TV	TT5	TT3	
		31	RJ	GS12	GT16	To S32
		32	MJ	0	EG	
			CA	GB33		

		IA	GS		Equation Generation Subroutines
(S1)	0	TU	RA10	GS2	Address of next word in Expanded List to "u" of TP
	1	RA	RA10	GC6	Advance address in Expanded List by 1 in "u"
	2	TP	[30000]	TT6	Next word in Expanded List to temp 6
	3	TP	GC10	Q	Mask for op. code to Q
	4	QT	TT6	A	Indicator from op. code of temp 6 to op. code of A
(S2)	5	MJ	0	[30000]	Exit
	6	TP	II4	TT2	[TU A 30000] → temp 2
	7	TU	TT5	TT2	Callword from "u" of temp 5 to "u" of temp 2
	10	TV	RA3	TT2	Current relative address to "v" of temp 2
	11	RJ	SI	SI1	Store instruction in temp 2 in routine image
(S3)	12	MJ	0	[30000]	Common exit
	13	TP	II5	TT2	[TV A 30000] → temp 2
	14	SP	TT5	17	
	15	TU	A	TT2	Callword from "v" of temp 5 to "u" of temp 2
(S4)	16	MJ	0	GS10	
	17	TP	II5	TT2	[TV A 30000] → temp 2
(S5)	20	MJ	0	GS10	
	21	TP	II4	TT2	[TU A 30000] → temp 2
(S8)	22	MJ	0	GS10	
	23	TP	II4	TT2	[TU A 30000] → temp 2
	24	TU	TT5	TT2	Callword from "u" of temp 5 to "u" of temp 2
	25	TV	RA4	TT2	Relative constant callword to "v" of temp 2
(S6)	26	MJ	0	GS11	
	27	TP	II33	TT2	[AT Q Q] → temp 2
	30	TU	RA3	TT2	Current relative address to "u" of temp 2
(S7)	31	MJ	0	GS10	
	32	TP	II33	TT2	[AT Q Q] → temp 2
	33	TU	RA4	TT2	Relative constant callword to "u" of temp 2
(S9)	34	MJ	0	GS10	
	35	TP	II33	TT2	[AT Q Q] → temp 2
	36	TU	RA4	TT2	Relative constant callword to "u" of temp 2
	37	MJ	0	GS11	
		CA	GS40		

(S10)	40	IA	GS40		
	41	TP	II33	TT2	[AT Q Q] → temp 2
(S11)	42	MJ	0	GS10	
	43	TP	II7	TT2	[SP A 17] → temp 2
		TU	TT6	TT	Callword from "u" of temp 6 to "u" of temp 0
(S12)	44	MJ	0	GV34	
	45	TP	II7	TT2	[SP A 17] → temp 2
(S13)	46	MJ	0	GS11	
	47	TP	II32	TT2	[SA 30000 0] → temp 2
	50	SP	TT6	17	Callword from "v" of temp 6 to "u" of A
	51	TU	A	TT	"u" of A to "u" of temp 2
(S14)	52	MJ	0	GV34	
	53	TP	II32	TT2	[SA 30000 0] → temp 2
	54	SP	TT5	17	Callword from "v" of temp 5 to "u" of A
(S15)	55	MJ	0	GS112	
	56	TP	II7	TT2	[SP A 17] → temp 2
(S16)	57	MJ	0	GS50	
	60	TP	II45	TT2	[TN Q Q] → temp 2
(S17)	61	MJ	0	GS11	
	62	TP	II1	TT2	[TP 30000 A] → temp 2
(S18)	63	MJ	0	GS50	
	64	TP	II1	TT2	[TP 30000 A] → temp 2
(S19)	65	MJ	0	GS11	
	66	TP	II1	TT2	[TP 30000 A] → temp 2
(S20)	67	MJ	0	GV32	
	70	TP	II1	TT2	[TP 30000 A] → temp 2
(S21)	71	MJ	0	GS43	
	72	TP	II5	TT2	[TV A 30000] → temp 2
	73	SP	TT5	17	Callword from "v" of temp 5 to "u" of A
	74	TU	A	TT2	"u" of A to "u" of temp 2
(S22)	75	MJ	0	GS25	
	76	TP	II10	TT2	[SA 30000 17] → temp 2
	77	TU	TT5	TT2	Callword from "u" of temp 5 to "u" of temp 2
		CA	GS100		

		IA	GS100	
Ⓢ23	100	MJ	0	GS11
	101	TP	II5	TT2
	102	MJ	0	GS7
Ⓢ24	103	TP	II4	TT2
	104	TV	RA3	TT2
				[TV A 30000] → temp 2
	105	MJ	0	GS111
Ⓢ25	106	TP	II4	TT2
	107	TU	TT6	TT2
				[TU A 30000] → temp 2
				Current relative address to "v" of temp 2
	110	MJ	0	GS10
	111	SP	TT6	17
	112	TU	A	TT2
	113	MJ	0	GS11
Ⓢ26	114	TP	II32	TT2
	115	MJ	0	GS77
		CA	GS116	
				Callword from "v" of temp 6 to "u" of A
				"u" of A to "u" of temp 2
				[SA 30000 0] → temp 2

		IA	GT		Generator Subroutines (cont.)
Ⓢ24	0	TP	II5	TT2	[TV A 30000] → temp 2
	1	MJ	0	GS24	
Ⓢ28	2	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2 To S29
	3	RJ	GT13	GT5	
	4	MJ	0	GS11	
Ⓢ29	5	TV	TT5	TT1	Callword from "v" of temp 5 to "v" of temp 1
Ⓢ30	6	TP	TT1	A	Callword from "v" of temp 1 to "v" of A
	7	TJ	GC34	GT11	Is callword partial result symbol? (i.e.,30---)
	10	MJ	0	GT12	No
	11	RJ	TR	TR2	Yes; pertinent temporary storage callword to "v" of A
	12	TV	A	TT2	Operand or temp callword to "v" of temp 2
	13	MJ	0	[30000]	Exit
Ⓢ31	14	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
	15	MJ	0	GS11	
Ⓢ32	16	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
Ⓢ32A	17	RJ	GT27	GT21	
	20	MJ	0	GS11	
Ⓢ33	21	TU	TT5	TT	Callword from "u" of temp 5 to "u" of temp 0
Ⓢ34	22	TP	TT	A	Callword from "u" of temp 0 to "u" of A
	23	TJ	GC35	GT25	Is callword partial result symbol? (i.e.,30---)
	24	MJ	0	GT26	No
	25	RJ	TR	TR1	Yes, pertinent temporary storage call- word to "u" of A
	26	TU	A	TT2	Operand or temp callword to "u" of temp 2
	27	MJ	0	[30000]	Exit
Ⓢ35	30	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
	31	TU	TT4	TT2	"u" of register indicator to "u" of temp 2
	32	MJ	0	GS11	
Ⓢ36	33	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
	34	RJ	GT13	GT5	
	35	MJ	0	GT31	
Ⓢ37	36	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
	37	TV	TT4	TT2	"v" of register indicator to "v" of temp 2
		CA	GT40		

		IA	GT40		
	40	MJ	0	GT17	
Ⓢ38	41	TU	TT5	TT3	Callword from "u" of temp 5 to "u" of temp 3
Ⓢ39	42	TV	TT5	TT3	Callword from "v" of temp 5 to "v" of temp 3
Ⓢ39A	43	RJ	SI	SI11	Inst. in temp 3 to relative constant image
	44	TP	II	TT2	[00 30000 30000] → temp 2
	45	MJ	0	GS11	
Ⓢ40	46	TU	TT5	TT3	Callword from "u" of temp 5 to "u" of temp 3
	47	MJ	0	GT43	
Ⓢ41	50	TV	TT4	TT3	"v" of register indicator to "v" of temp 3
	51	MJ	0	GT46	
Ⓢ42	52	TU	TT4	TT3	"u" of register indicator to "u" of temp 3
	53	MJ	0	GT42	
Ⓢ43	54	TP	TT3	TT2	[F 30000 30000] from temp 3 → temp 2
	55	RJ	GT13	GT5	To S29
	56	RJ	GT27	GT21	To S33
	57	MJ	0	GS11	
Ⓢ44	60	TU	TT5	TT3	Callword from "u" of temp 5 to "u" of temp 3
	61	RJ	GT13	GT5	To S29
	62	TV	A	TT3	Callword of variable, constant or temp to "v" of temp 3
	63	MJ	0	GT43	
Ⓢ45	64	TV	TT5	TT3	Callword from "v" of temp 5 to "v" of temp 3
	65	RJ	GT27	GT21	To S33
	66	TU	A	TT3	Callword of variable, constant or temp to "u" of temp 3
	67	MJ	0	GT43	
Ⓢ46	70	TP	TT3	TT2	[F_ 30000 30000] from temp 3 → temp 2
	71	TV	TT4	TT2	"v" of register indicator to "v" of temp 2
	72	MJ	0	GS11	
Ⓢ47	73	TP	II6	TT2	[RA 30000 30000] → temp 2
	74	TU	RA3	TT2	Current relative address to "u" of temp 2
	75	MJ	0	GV4	
		CA	GT76		

S49	0	IA RA	GU RA3	GC3	Generator Subroutine (Int. Power) Advance current rel. address by 1 in "u" and "v"
	1	TP	II36	TT2	[FD 30000 Q] → temp 2
	2	TU	TT6	TT2	Callword from "u" of temp 6 to "u" of temp 2
S50	3	MJ	0	GS11	
	4	TP	II35	TT2	[FM Q 30000] → temp 2
	5	TV	RA6	TT2	Current reusable temp callword to "v" of temp 2
	6	RS	RA6	GC3	Decrease current reusable temp callword by one
S51	7	MJ	0	GS11	
	10	TP	II22	TT2	[FM 30000 30000] → temp 2
	11	TU	RA6	TT2	Current reusable temp callword to "u" of temp 2
	12	TV	RA6	TT2	Current reusable temp callword to "v" of temp 2
S52	13	MJ	0	GS11	
	14	TP	II1	TT3	[TP 30000 A] → temp 3
	15	TU	TT5	TT3	Callword from "u" of temp 5 to "u" of temp 3
	16	RJ	GK34	GK30	Callword of available reusable temp to "v" of temp 3
S53	17	MJ	0	GT43	
	20	TP	II1	TT3	[TP 30000 A] → temp 3
	21	RJ	GK34	GK30	Callword of available reusable temp to "v" of temp 3
	22	TP	TT3	TT2	Instruction from temp 3 to temp 2
S54	23	MJ	0	GS11	
	24	TP	II10	TT2	[SA 30000 17] → temp 2
	25	TU	RA4	TT2	Relative constant callword to "u" of temp 2
S55	26	MJ	0	GS11	
	27	TU	TT5	TT	Callword from "u" of temp 5 to "u" of temp 0
	30	SP	TT	25	
	31	LT	0	TT3	Callword from "u" of temp 0 to "v" of temp 3
	32	RJ	SI	SI11	Inst. in temp 3 to relative constant image
S56	33	MJ	0	[30000]	Exit
	34	TP	II22	TT2	[FM 30000 30000] → temp 2
	35	RJ	GT27	GT21	To S33
	36	SP	TT2	25	
	37	LT	0	A	
		CA	GU40		

	40	IA TV	GU40 A	TT2	Callword from "u" of temp 2 to "v" of temp 2
Ⓢ57	41	MJ	0	GS11	
	42	TP	II27	TT2	[RP 30000 30000] → temp 2
	43	TP	GC12	Q	Mask for rightmost "two" octal digits of "v" → Q
	44	QT	TT6	A	Exponent from temp 6 to "v" of A
	45	SS	GC13	17	Exponent less two to "u" of A
	46	TU	A	TT2	jn to "u" of repeat instruction in temp 2
	47	TV	RA3	TT2	Current relative address to "v" of RP inst. in temp 2
	50	RA	TT2	GC11	Advance "v" of RP inst. in temp 2 by one
Ⓢ58	51	MJ	0	GS11	
	52	TP	II22	TT2	[FM 30000 30000] → temp 2
Ⓢ59	53	MJ	0	GS11	To ③
	54	TP	GC47	Q	Mask for op. code and "v" to Q
	55	QS	II32	TT2	[32 ___ 00000] to op. code and "v" of temp 2
	56	MJ CA	0 GU57	GS11	

		IA	GV		Generator Subroutines (Subscript Operator)
R1	0	TP	II25	TT2	[MA 30000 30000] → temp 2
	1	RS	GV2	GC6	Decrease "u" of next instruction by one
	2	TP	[30000]	TT6	Next subscript word from Expanded List to temp 6
	3	TU	TT6	TT2	Callword of multiplier from "v" of temp 6 to "v" of temp 2
R3	4	TV	TT6	TT1	Callword of subscript from "v" of temp 6 to "v" of temp 1
	5	RJ	GT13	GT6	To S30
	6	MJ	0	GS11	
R2	7	TP	II24	TT2	[MP 30000 30000] → temp 2
	10	MJ	0	GV1	
R4	11	TP	II32	TT2	[SA 30000 0] → temp 2
	12	RS	GV2	GC6	Decrease by one in "u"
	13	TU	A	GV14	Preset address of next subscript word in Expanded List
	14	TP	[30000]	TT6	Next subscript word from Expanded List to temp 6
	15	SP	TT6	17	
	16	TU	A	TT	Callword of subscript from "v" of temp 6 to "u" of temp 0
	17	RJ	GT27	GT22	To S34
	20	TP	TT2	TT3	Generated Instruction to temp 3
	21	MJ	0	[30000]	
R5	22	TP	II15	TT2	[TJ 30000 30000] → temp 2
	23	TV	RA3	TT2	Current relative address to "v" of temp 2
	24	RA	TT2	GC11	Advance "v" of temp 2
	25	TU	TT5	TT2	Callword from "u" of temp 5 to "u" of temp 2
	26	MJ	0	GS11	
R6	27	TP	II26	TT2	[DV 30000 Q] → temp 2
	30	MJ	0	GV25	
R7	31	TP	II1	TT2	[TP 30000 A] → temp 2
	32	SP	TT5	17	Callword from "v" of temp 5 to "u" of A
	33	TU	A	TT	Callword from "u" of A to "u" of temp 0
	34	RJ	GT27	GT22	To S34
	35	MJ	0	GS11	
S48	36	TP	II7	TT2	[SP A 17] → temp 2
	37	MJ	0	GV32	
		CA	GV40		

Store inst. in Temp 2 in Routine Image	X1	0	IA	SI		Store inst. in Routine Image or Relative Constant Image
		1	MJ	0	[30000]	Exit
		2	TP	TT2	[30000]	Inst. in temp 2 to current address in Routine Buffer
		3	RA	SI1	GC11	Advance current address in routine buffer by one
		4	TJ	GC25	SI	Are there 170g words in routine buffer? Exit if no.
		5	TV	GC24	SI1	Yes; reset current address in routine buffer to initial address
		6	RP	30170	SI7	Transfer 170g generated Inst. from routine buffer to current address in routine image on drum
		7	TP	RB	[30000]	
		8	RA	SI6	GC26	Advance current address in routine image by 170g
		9	MJ	0	SI	To Exit
Store Rel. Const. in Const. Image	X2	10	RA	RA4	GC3	Advance current relative constant callword by one
		11	TP	TT3	[30000]	Relative constant (or "10" line) in temp 3 to relative constant image
		12	RA	SI12	GC11	Advance current address in relative constant image by one
		13	TJ	LG	SI	Are there more than 77g relative constants? Exit if no.
		14	RJ	WA	WA1	Yes, type SENTENCE___ (equation)
		15	TP	TO	UP3	Parameter for alarm text to type routine
		16	RJ	UP2	UP	Type: SENTENCE TOO LONG.
		17	MJ	0	BQ6	Rewind tapes and stop
		18	CA	SI21		

Y1
Y2

	IA	TR		
0	MJ	0	[30000]	Obtain Redundancy or Reusable Temp Callword for Partial Result
1	LT	25	A	Exit
2	RP	[30000]	TR12	Partial result symbol to "v" of A Is partial result symbol in Redundant Partial Result List?
3	EJ	RL	TR4	Yes; to TR4. No; to TR12
4	SP	Q	17	jn - r → "u" of A
5	AT	Q	Q	jn - r → "u" & "v" of Q
6	TP	RA2	A	jn → "u" & "v" of A
7	SS	Q	0	+r → "u" & "v" of A
10	SA	RA5	0	Callword of redundancy temp for partial result to "u" and "v" of A
11	MJ	0	TR	To exit
12	RS	RA6	GC3	Decrease current reusable temp callword by one
13	SA	GC3	0	Callword of reusable temp for partial result to "u" and "v" of A
14	MJ	0	TR	To exit
	CA	TR15		

	IA	GC		Generator Constants (Fixed)
0	0	4	4	
1	0	3	3	
2	0	2	2	
3	0	1	1	
4	0	5	5	
5	0	6	6	
6	0	1	0	
7	0	0	7	
10	77	0	0	
11	0	0	1	
12	0	0	77	
13	0	0	2	
14	02	0	0	
15	0	26000	0	
16	0	RI	FL	Parameter to write generated routine from drum to tape
17	0	0	0	
20	0	0	RB7	
21	0	0	GK	To set switch (A)
22	0	0	ZZ24	
23	0	00777	00777	
24	TP	TT2	RB	Initial address in routine buffer in "v"
25	TP	TT2	RB170	
26	0	0	170	
27	TP	RB	RI	Initial address in routine image on drum in "v"
30	TP	TT3	CI	Initial address in relative constant image in "v"
31	0	0	30000	
32	10	0	3	
33	0	0	EG	To set switch (A) and (B)
34	0	0	31000	
35	0	31000	0	
36	0	2	0	
37	0	3	0	
	CA	GC40		

	IA	GC40	
40	0	0	6
41	0	0	GK3
42	TP	TT2	RB171
43	0	RB	FL
44	0	1000	1000
45	0	20000	20000
46	0	23000	0
47	77	0	77777
50	0	10000	10000
51	0	57777	57777
52	0	67777	67777
53	0	776	776
	CA	GC54	

To switch (B)

Parameter to write generated routine
from core to tape

Initial relative constant callword
Initial redundancy temp callword less 1
Initial reusable temp callword less 1

	IA	II	
0	00	30000	30000
1	11	30000	A
2	12	30000	Q
3	13	30000	Q
4	15	A	30000
5	16	A	30000
6	21	30000	30000
7	31	A	17
10	32	30000	17
11	34	30000	30000
12	35	30000	A
13	36	30000	A
14	37	50051	50051
15	42	30000	30000
16	45	0	0
17	56	0	30000
20	64	30000	30000
21	65	30000	30000
22	66	30000	30000
23	67	30000	30000
24	71	A	30000
25	72	30000	30000
26	73	30000	Q
27	75	30000	30000
30	11	30000	50051
31	13	A	A
32	32	30000	0
33	35	Q	Q
34	66	A	A
35	66	Q	30000
36	67	30000	Q
37	73	30000	A
	CA	II40	
	IA	II40	
40	64	Q	30000
41	11	Q	A
42	11	A	30000
43	10	2	0
44	12	A	A
45	13	Q	Q
46	11	30000	30000
47	45	0	01000
50	45	0	30000
	CA	II51	

Dummy Instructions

Callword of "Square Root" Library
Routine (SQRT) in "u" and "v"

Callword of "Square Root" Library
Routine (SQRT) in "v"

	IA	TO		Alarm Text
0	40	T01	3	Parameter for alarm text
1	65	30506	63050	S E N T E N
2	26	30016	65151	C E Δ T O O
3	01	46515	03222	Δ L O N G .
	CA	T04		
	IA	LG		Limiting Values
0	TP	TT3	CI100	Maximum address in relative constant image + 1
1	0	1002	1002	} Maximum number of lines in object program body (including jump to exit) + 1
	CA	LG2		

Explanation of Relative (Running) Address List

RAO	—	—	—	jn for "Q" List search in "u"
1	—	—	—	jn for "A" List search in "u"
2	—	—	—	jn for Redundant Partial Result List
3	—	—	—	Current relative object program address in "u" and "v"
4	—	—	—	Current relative constant callword in "u" and "v"
5	—	—	—	Initial redundancy temp callword less 1 in "u" and "v"
6	—	—	—	Current reusable temporary storage callword in "u" and "v"
7	—	—	—	Highest reusable temporary storage callword in "u" and "v"
10	—	—	—	Initial address in Expanded List + 2 in "u"

Explanation of Working Temporaries (TT)

TT0	0	[30000]	0	Temp 0 - op. code and "v" always zero
1	0	0	[30000]	Temp 1 - op. code and "u" always zero
2	0	0	0	Temp 2 - usually generated instruction to be stored
3	0	0	0	Temp 3 - usually relative constant to be stored
4	0	0	0	Temp 4 - register indicator
5	0	0	0	Temp 5 - operator word from Expanded List
6	0	0	0	Temp 6 - indicator word from Expanded List
7	0	0	0	Temp 7 - index counter
10	0	0	0	Temp 10 - "10" line counter

V. ALLOCATION PHASE

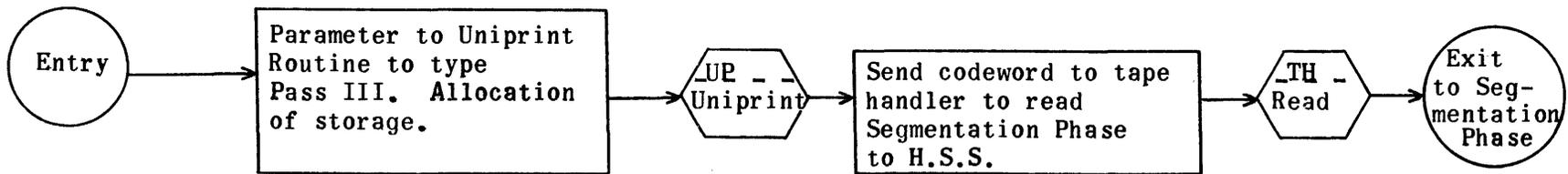
V. ALLOCATION PHASE

1. Segmentor

a. Segmentation Setup

This Setup Routine for Segmentation prints out the information that the Allocation Phase (including the Segmentor, the Allocator, and the Initialization Generator) is about to begin.

The routine reads the nine blocks of Segmentation from the UNICODE Master Tape and then jumps into the phase.



Flow Chart for Set-Up Segmentation

Segmentation Set-Up Regions

RE	ZS7230	Loading address for segmentation setup
RE	ZZ7230	Operating address of segmentation setup
RE	SA674	Loading and operating address of segmentation phase
RE	SL1100	11 = number of blocks for segmentation phase
RE	TH21	
RE	UP421	

Set-Up for Segmentor

	IA	ZS		
ZZ0	TP	ZZ6	UP3	} Print: Pass III. ALLOCATION OF STORAGE.
1	RJ	UP2	UP	
2	TP	ZZ5	TH3	} Read Segmentation Phase to H.S.S.
3	RJ	TH2	TH	
4	MJ	0	SA	Enter Segmentation Phase.
5	50	SL1	SA	Tape handler code word (Read).
6	0	ZZ7	7	Code word for UP
7	01	01010	10101	△ △ △ △ △ △
10	52	24656	50134	P A S S △ I
11	34	34220	10101	I I . △ △ △
12	01	24464	65126	△ A L L 0 C
13	24	66345	15001	A T I 0 N △
14	51	31016	56651	0 F △ S T 0
15	54	24323	02201	R A G E . △
	CA	ZS16		

b. Segmentation

Phase I.

Phase I prepares two directories using Op File I of the generated routines on Uniservo 5 and Op File I of the library routines on Uniservo 2. First, all items of Op File I on the generated routine tape are read into H.S.S. and then transferred to the MD. Directory I is constructed by making an entry for each item placed on the MD. The first word of this entry contains the call word for this item in the u position; the second contains the locating MD address for this item in the y position.

When Op File I of the generated routine tape has been completely read into H.S.S., List 1 (a listing of all library routines required for the problem prepared during translation) is read into H.S.S. (List 1 is stored following Op File I of the generated routine tape.) Next Op File I of the library tape is read from tape and checked for the occurrence of the items of List 1. When an item of List 1 is found in the library Op File I, the Op File for this item is placed on the MD and an entry is made in Directory 1.

Directory 2 consists of only two words. The first word holds the MD address of the first statement Op File; the second contains information relating to the MD address of the last statement Op File. This two word directory is prepared concurrently with Directory 1.

Phase II.

Phase II uses Directories 1 and 2 to divide the problem into efficient running segments producing Op File IIa and IIb on tape for each segment.

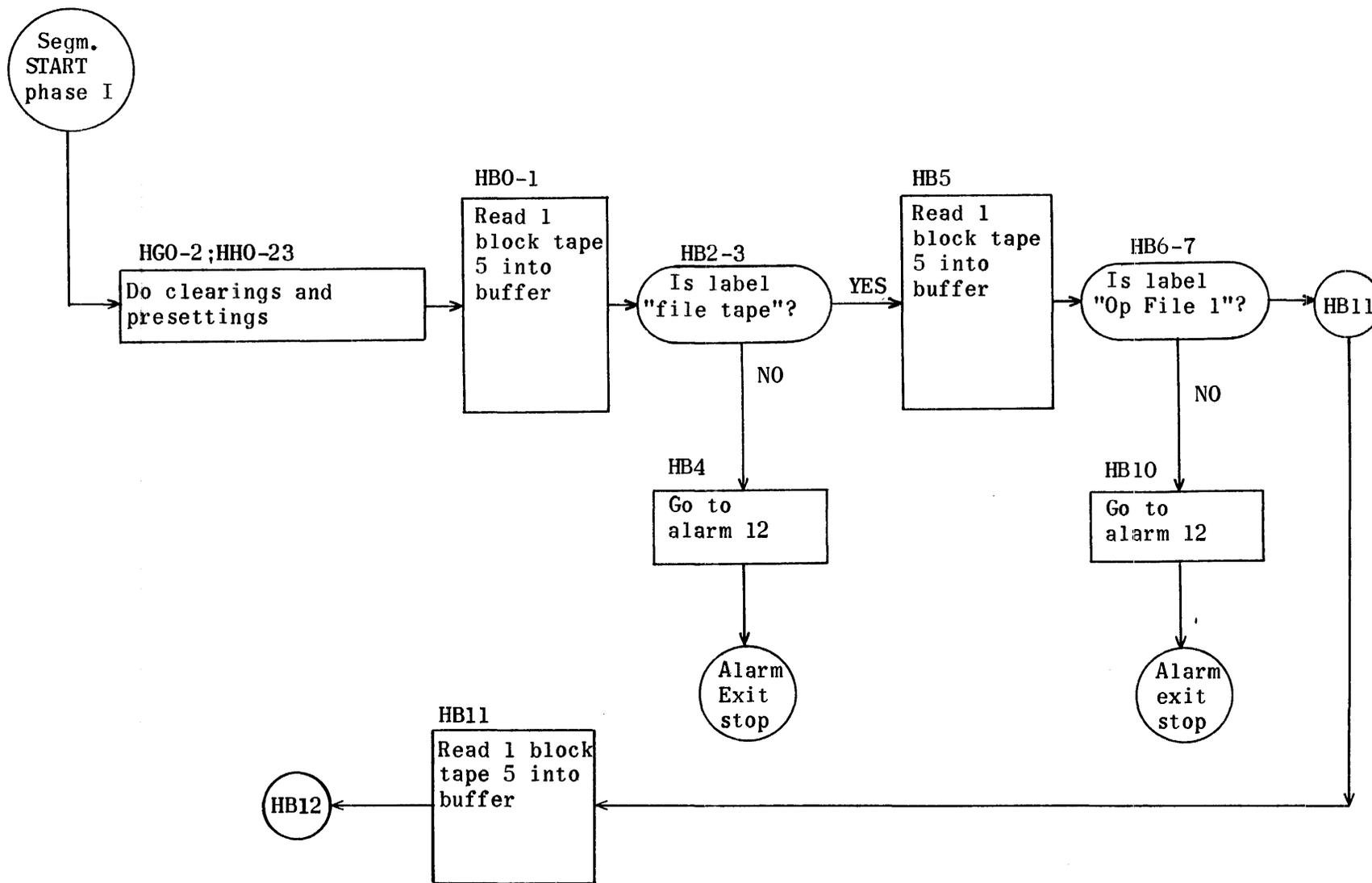
Using the first word of Directory 2 (location of Op File I for the first statement) as the initial point, Op File I for each statement is processed in sequence. A sub-tally of the total number of lines of coding required for each current statement and its necessary cross references is maintained. This in turn updates a master tally for the segment which contains the accumulated total number of lines needed for all statements and their required cross reference routines. After processing each complete statement, the master tally is checked to determine if it is within the prescribed limits ($4096m-N$; where N is the length of the Control Section and m the number of core banks available). If it has exceeded the set limits, the sub-tally is subtracted from the master tally and this becomes the length of the segment. If a single statement and its necessary cross references exceeds ($4096m-N$) the routine gives an alarm. The last statement processed which exceeds the set limit becomes the first entry in the following segment.

Vary loops are treated differently in order to avoid unnecessary jumping between segments. All statements within the range of a Vary are counted together in the sub-tally as one large statement, including other Vary statements that might be nested within the first loop. If the master tally then exceeds ($4096m-N$), the routines check whether the sub-tally exceeds ($4096m-N$). If not, the routine ends a segment right before the Vary statement, starting the next segment with a Vary loop. If the Vary loop in itself exceeds ($4096m-N$), the segmentation goes backward within the sentences of the Vary loop until the limit ($4096m-N$) is reached again. If there is no Vary

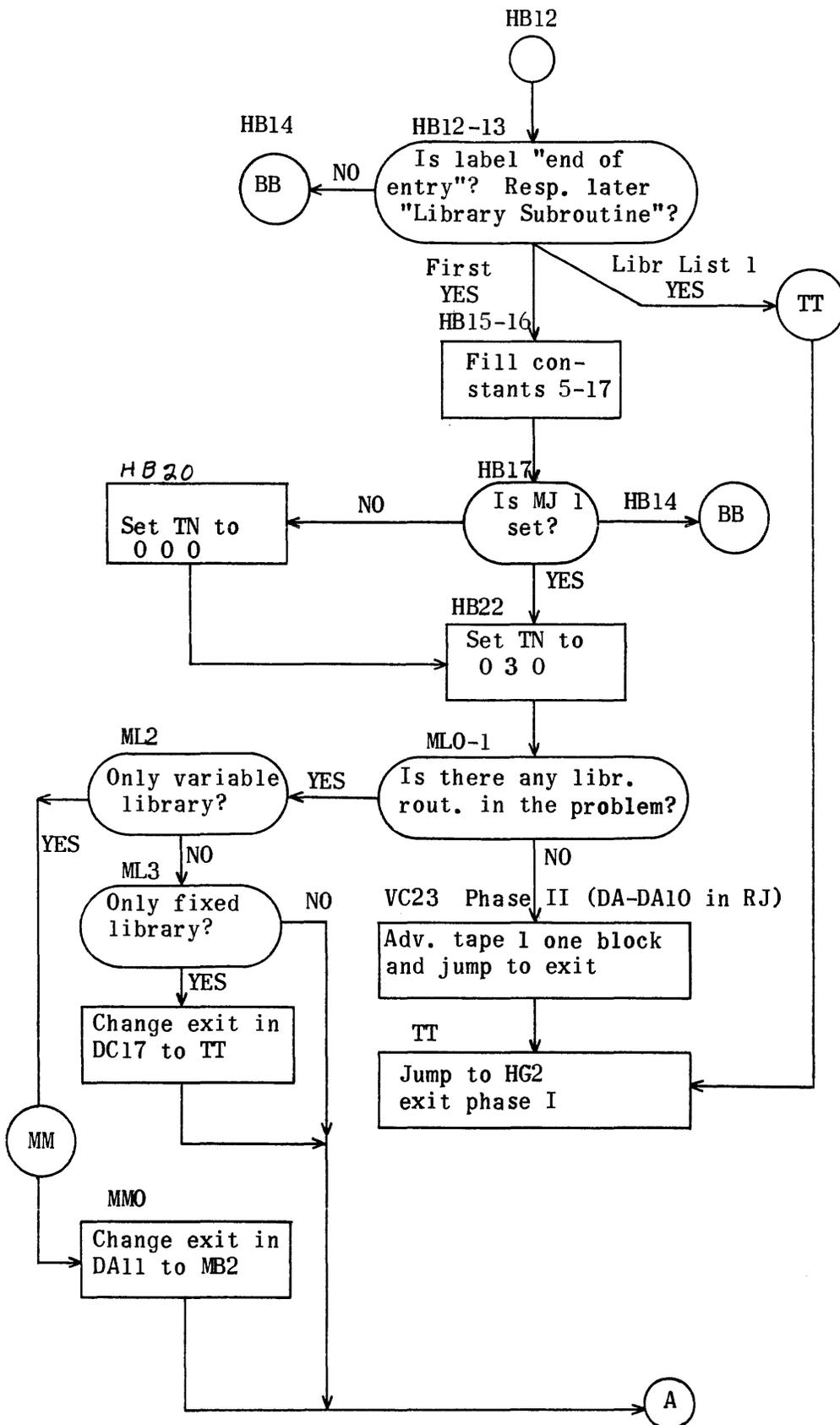
within the Vary, it forms a segment right there. If there are other Vary sentences nested within the large Vary loop, it goes further back beyond the next Vary statement and forms the segment so that the new segment would start with a Vary Statement.

Processing continues entering each item in turn into Op File IIa using the length (4096m-N) as a limit for each segment. Whenever cross references to other statements (open jumps) are recognized, these call words are entered into Op File IIb. Thus, Op File IIb is a listing of jump cross reference call words for each segment. When sufficient statements for one segment have been processed and their call words entered into Op File IIa and IIb (as needed), these files are written on tape ready for use by the allocator. The process is repeated, building Op File IIa and IIb for each segment using the second word of Directory 2 to indicate when the last statement in Op File I has been processed.

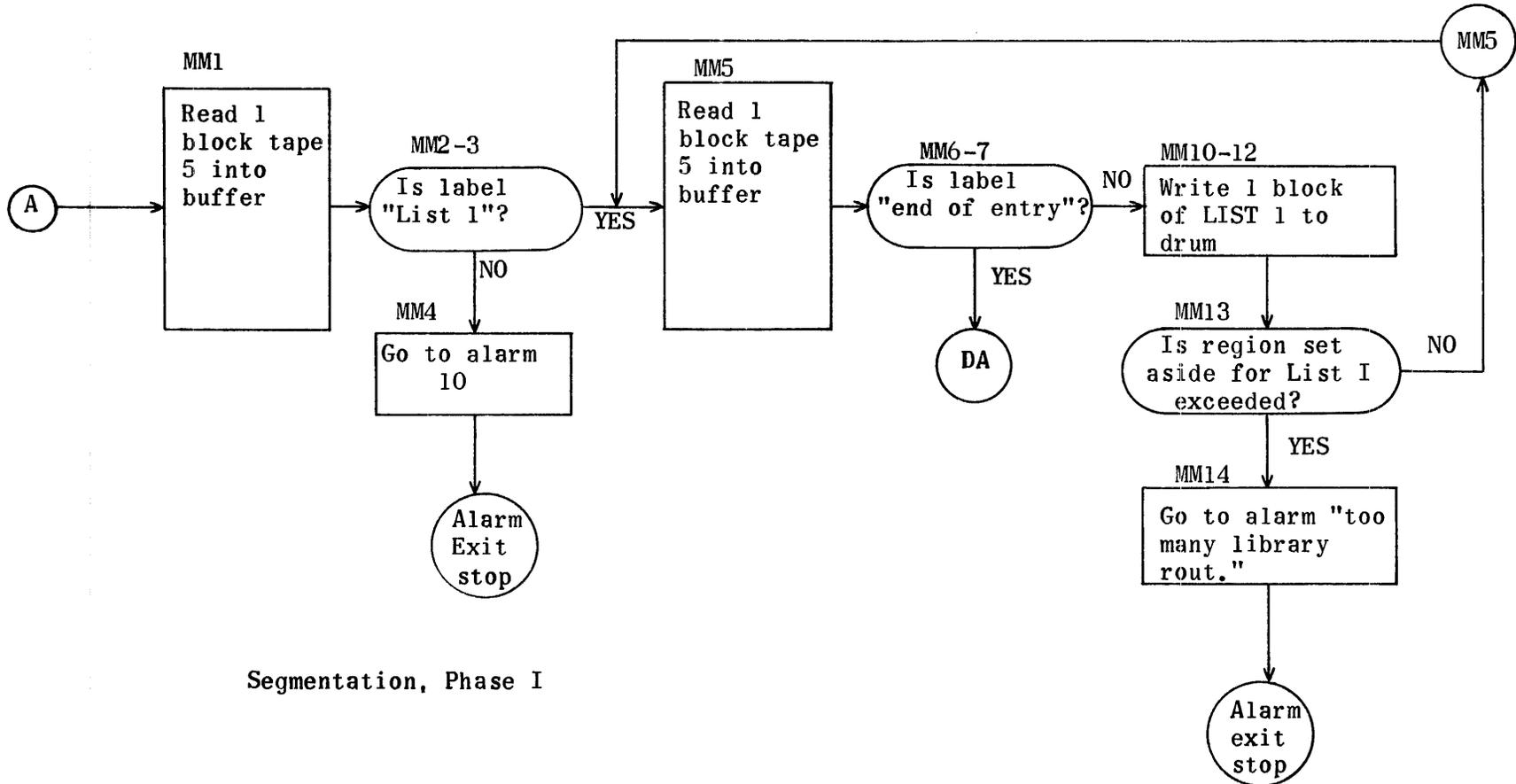
Segmentation, Phase I



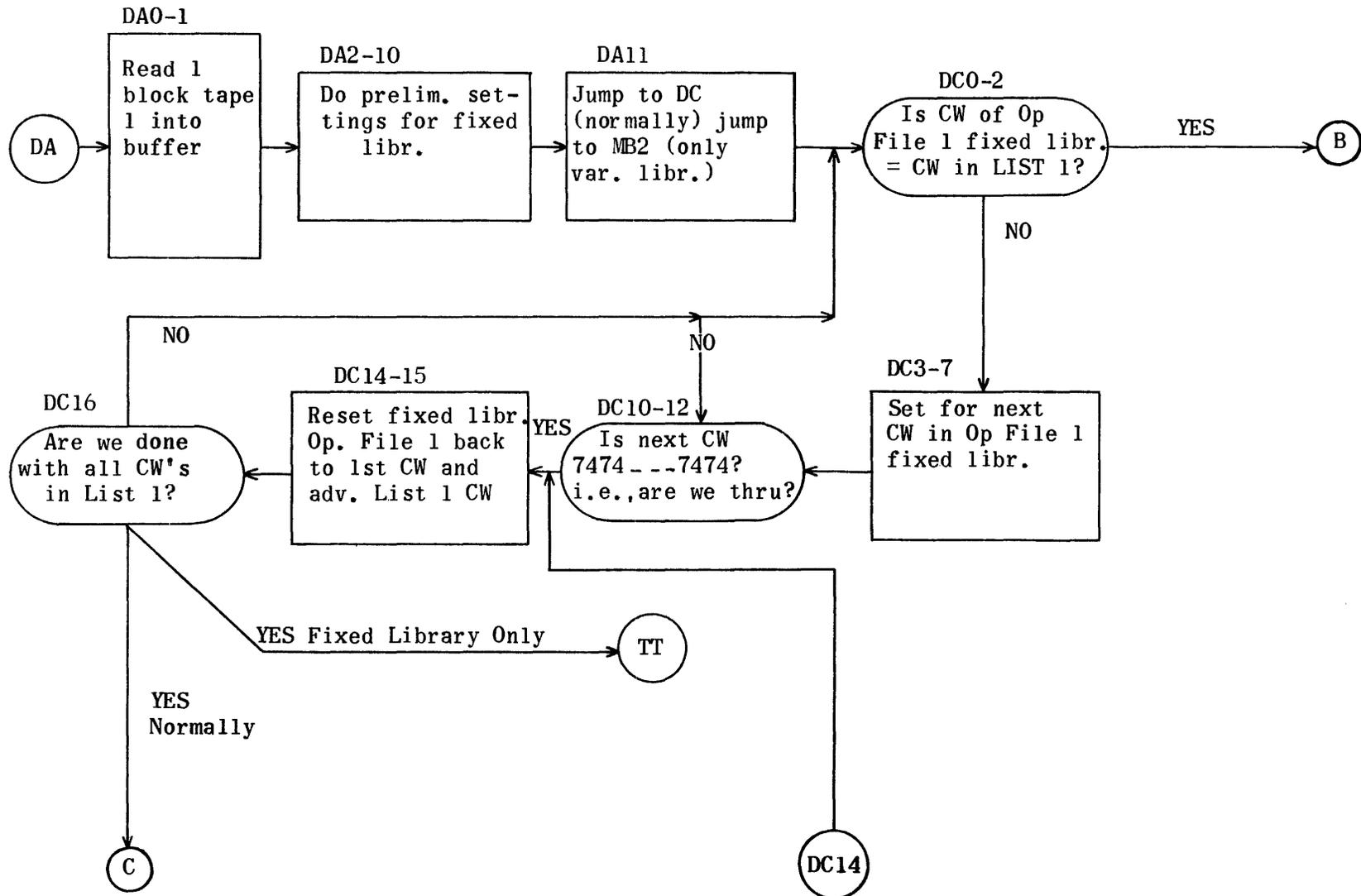
1467



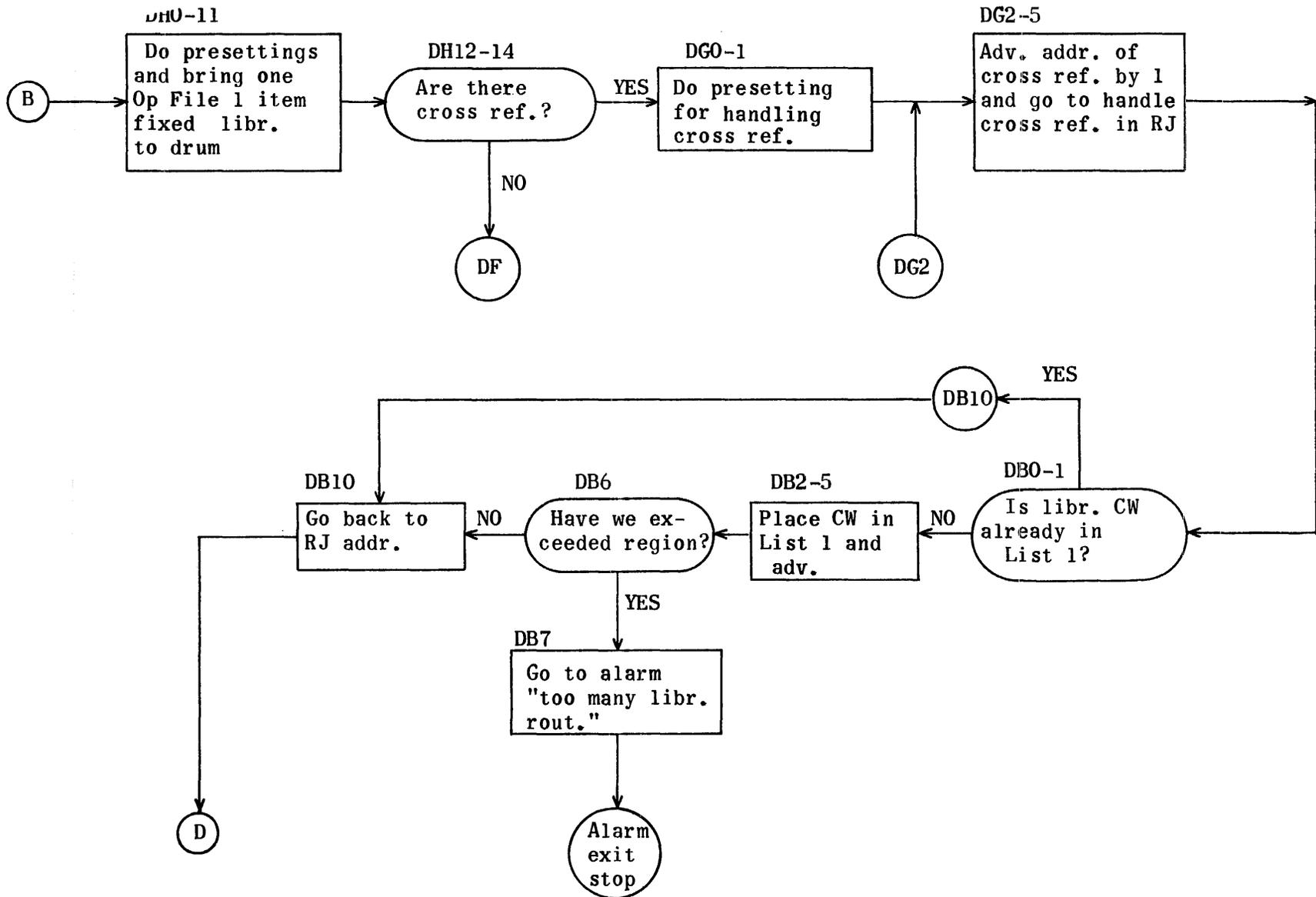
Segmentation, Phase I

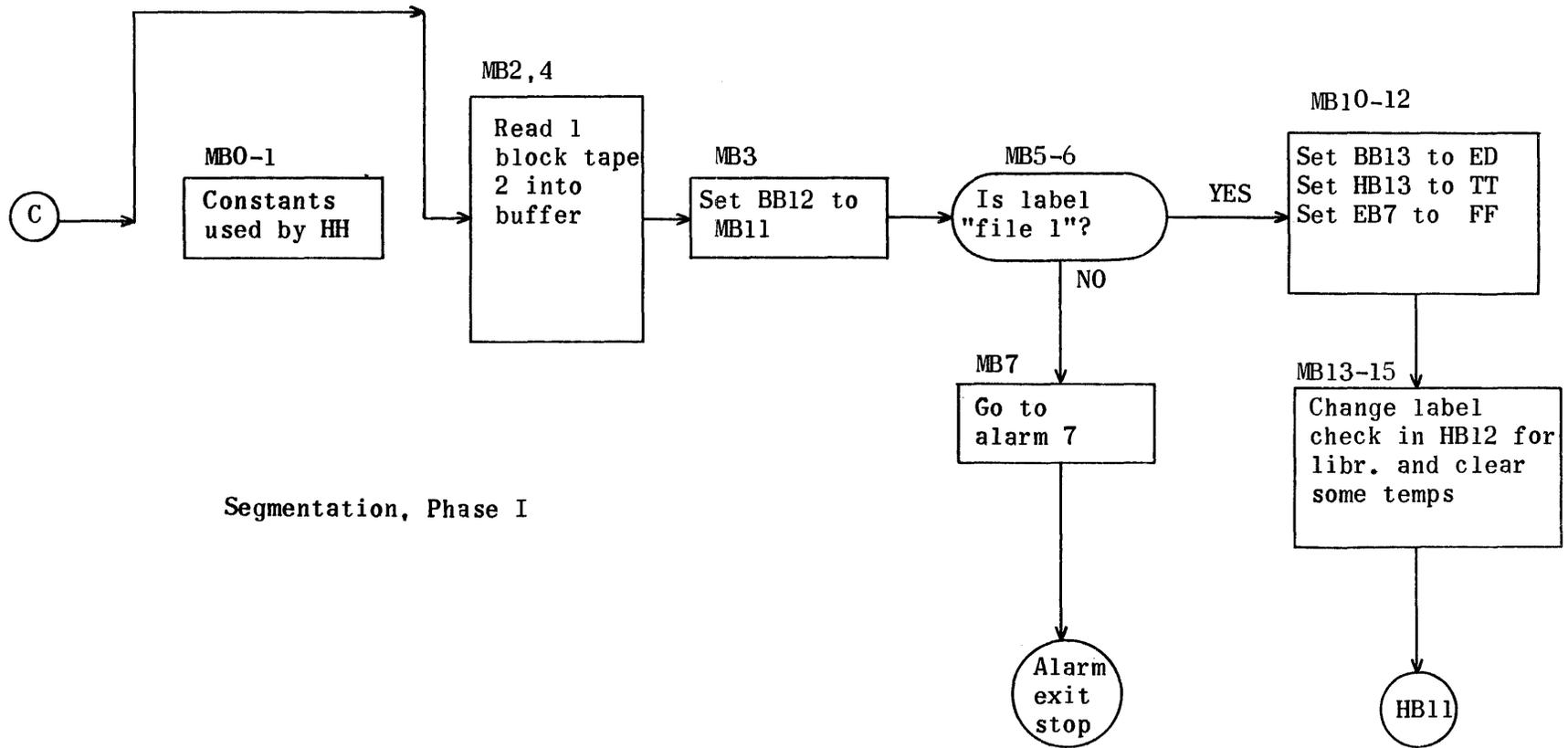


Segmentation, Phase I

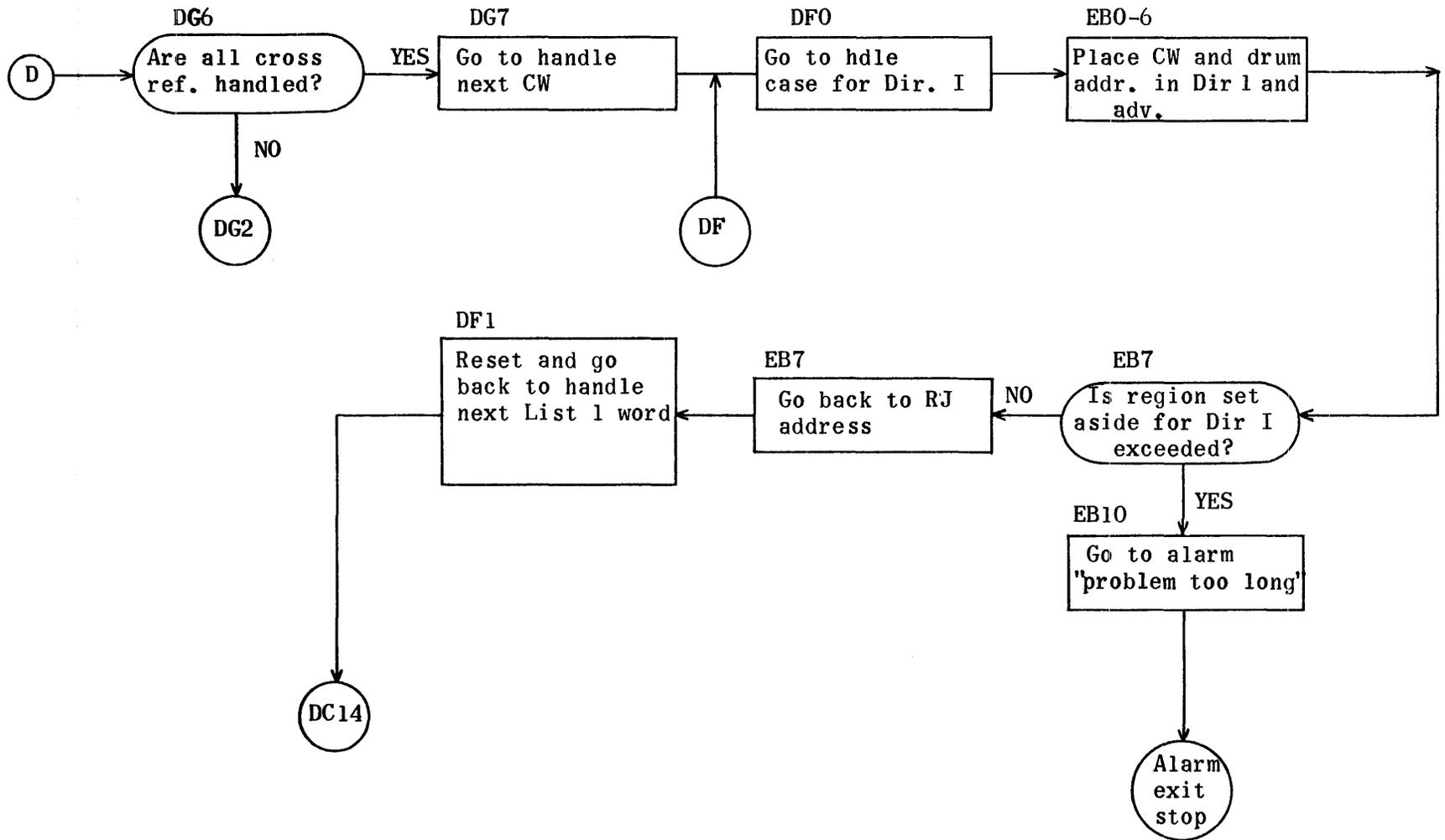


Segmentation, Phase I

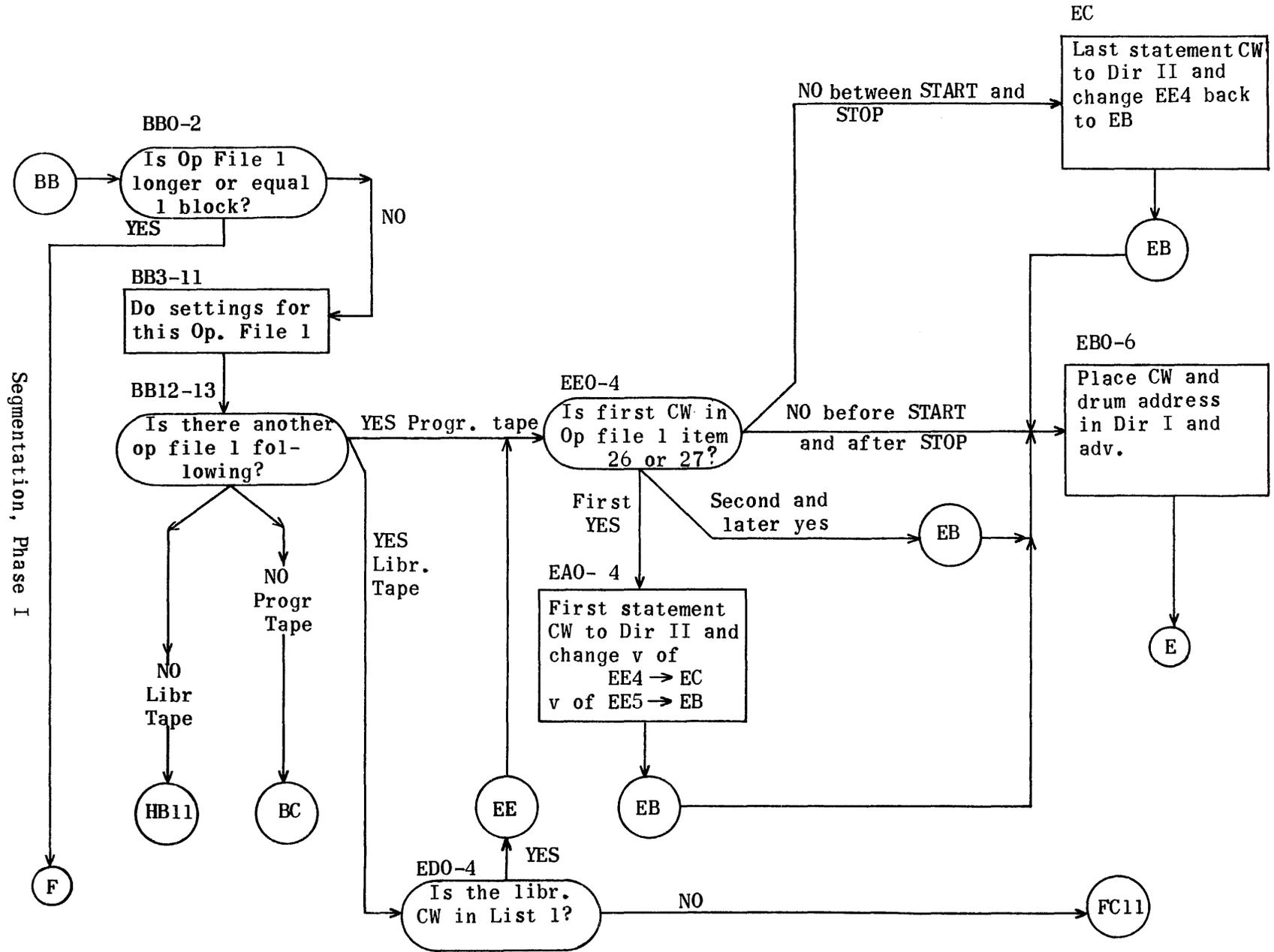




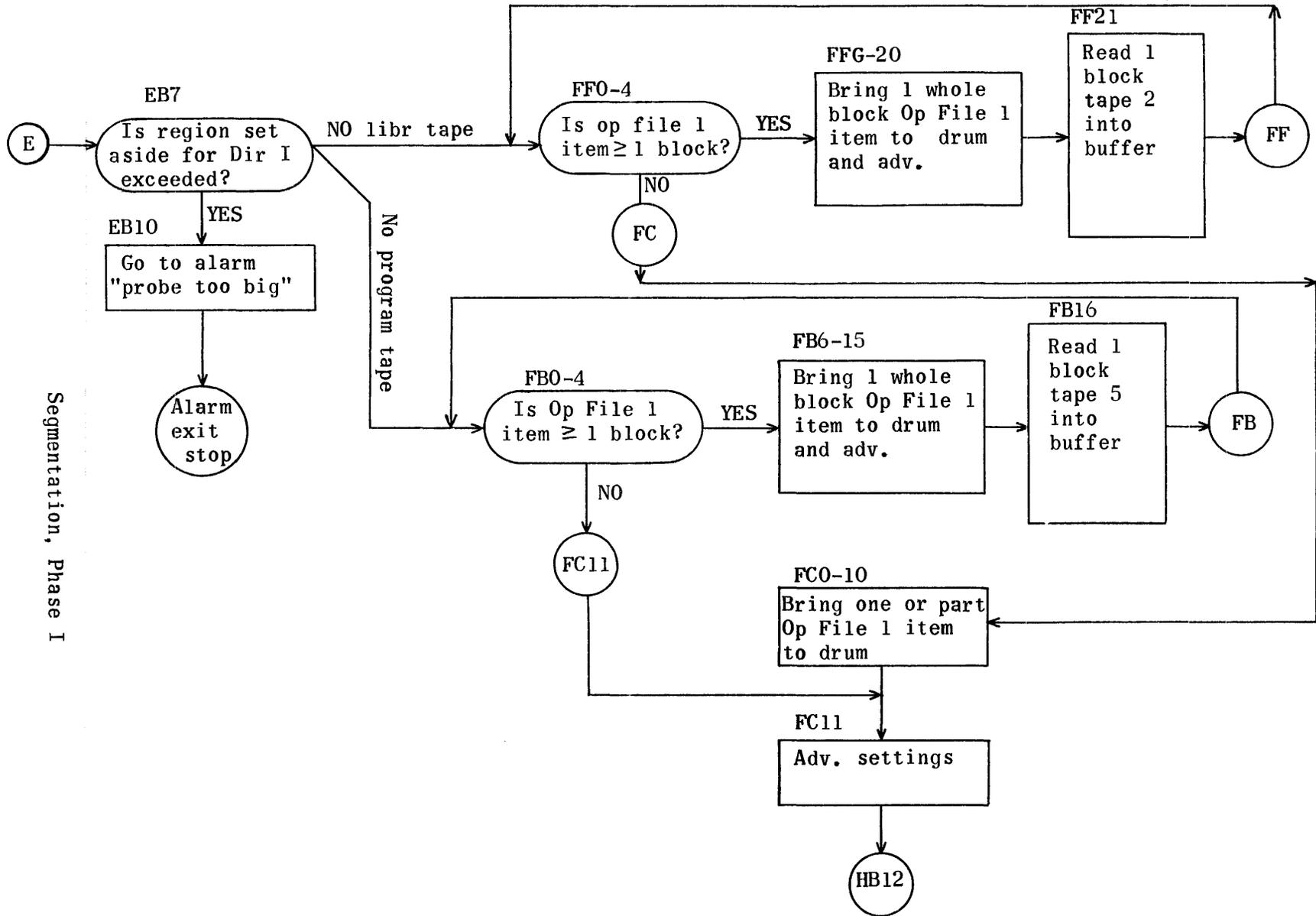
Segmentation, Phase I

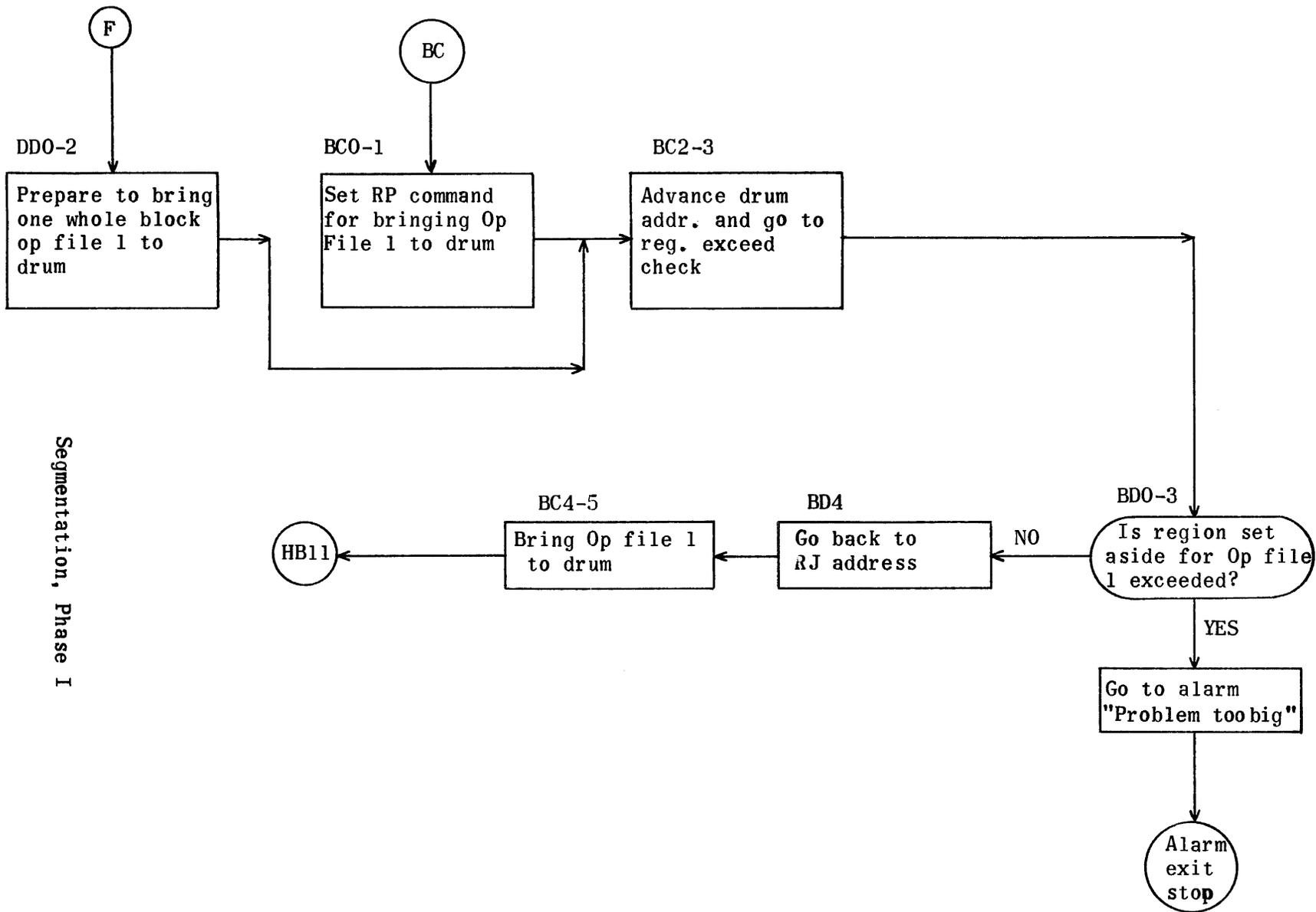


Segmentation, Phase I



Segmentation, Phase I

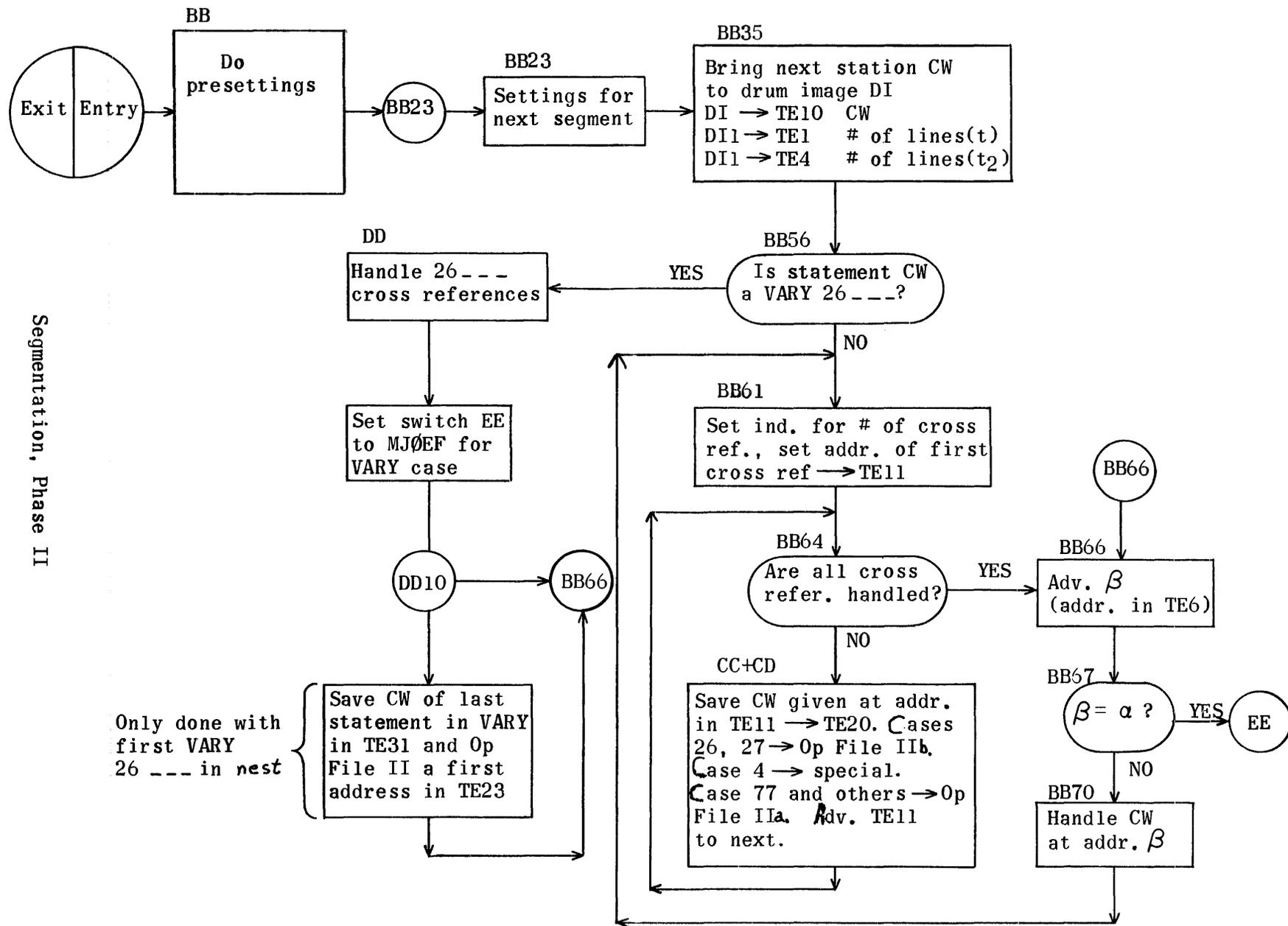


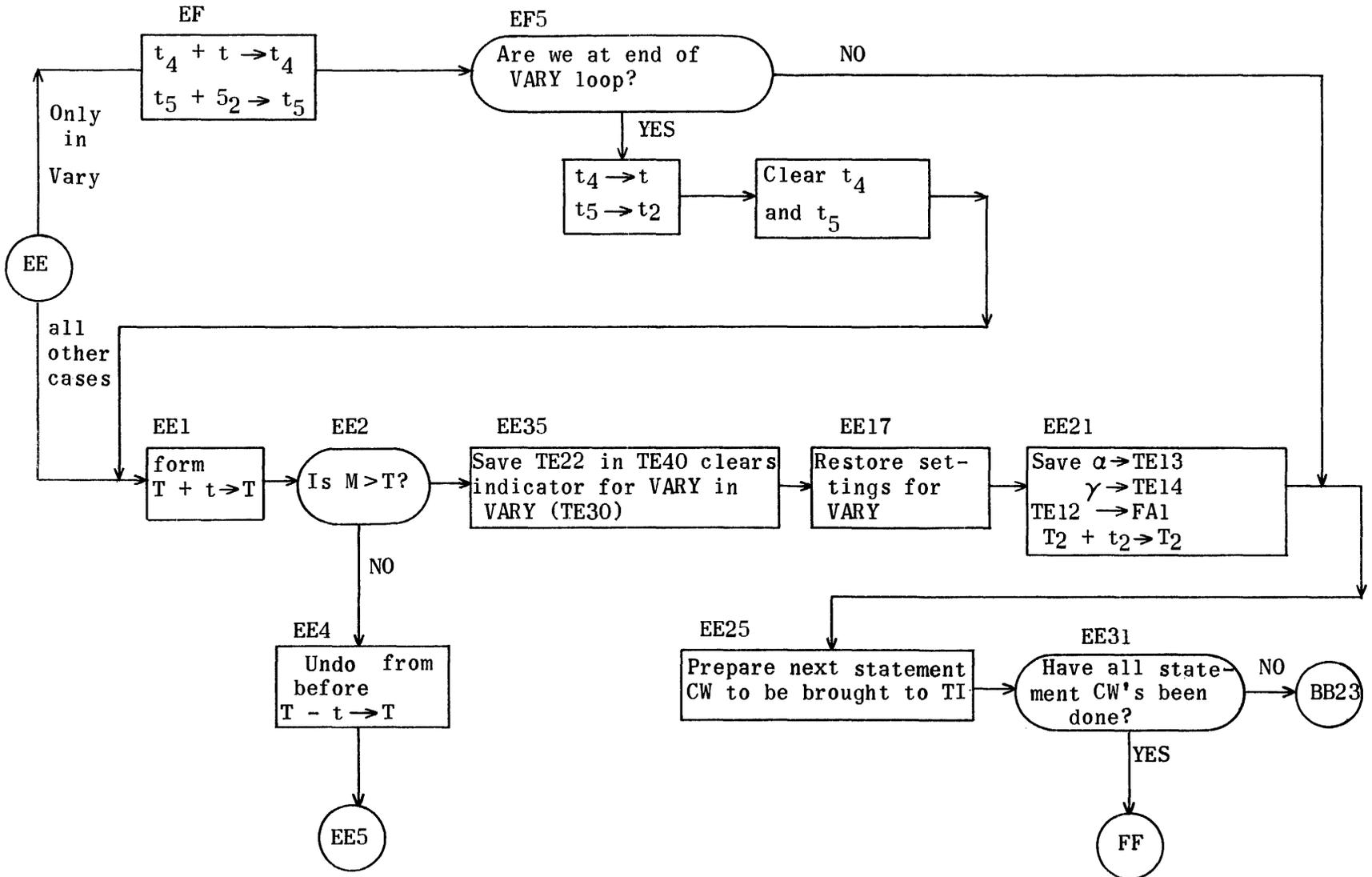


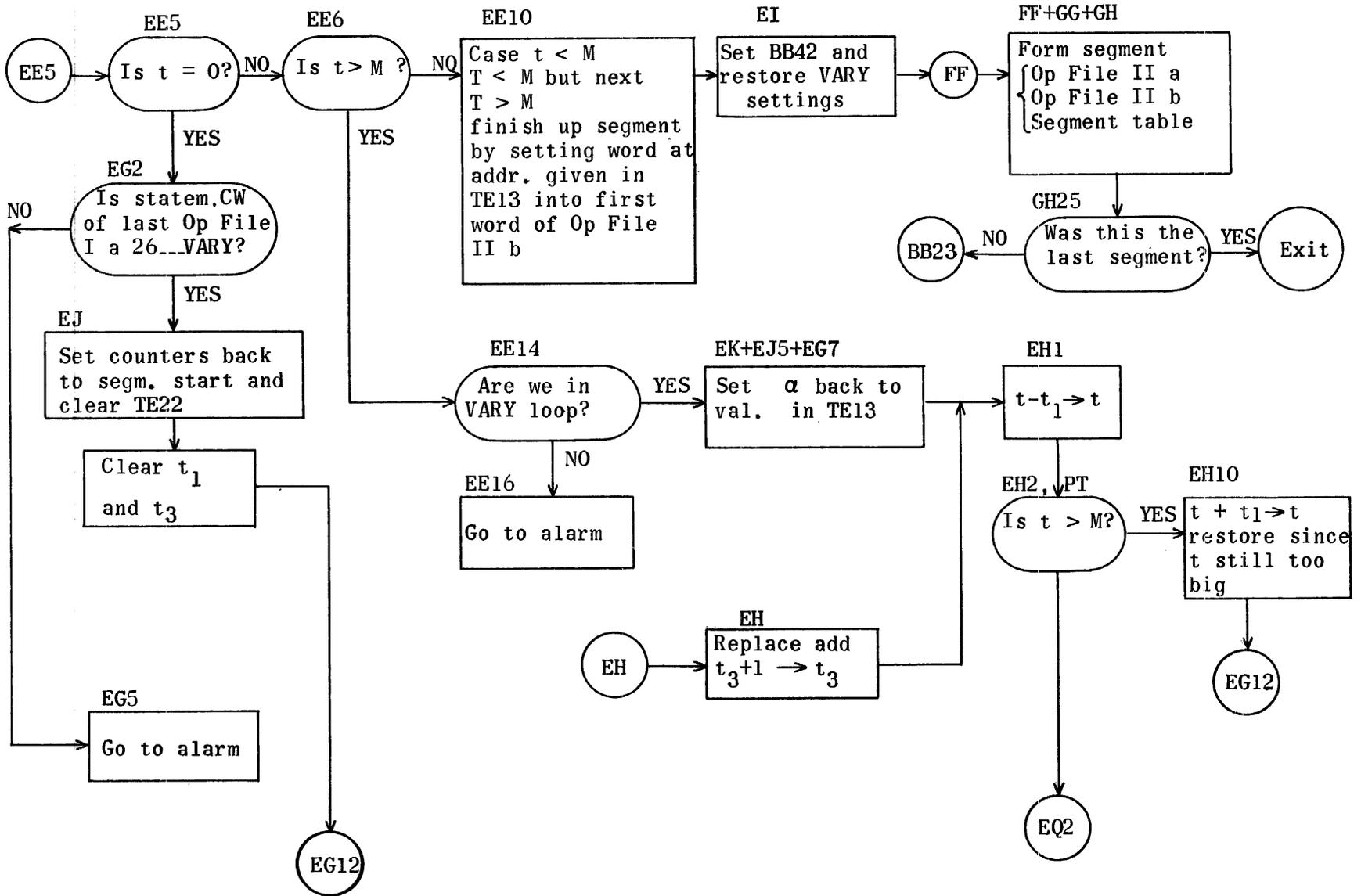
Segmentation, Phase I

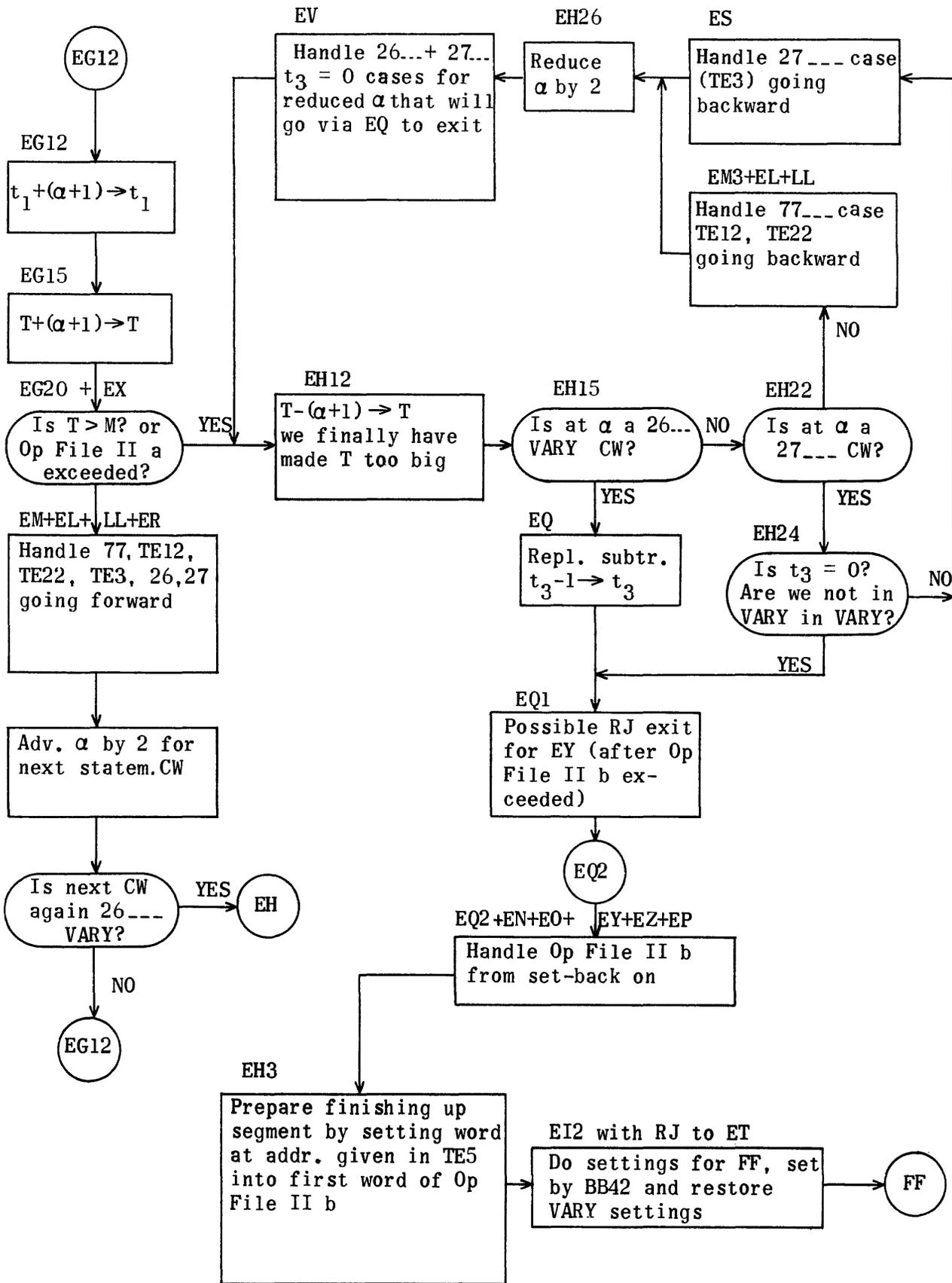
1476

Segmentation, Phase II









Segmentation, Phase II

Segmentation Regions Phase I

RE ZA77000	
RE BQ632	
RE BR537	
RE ST653	Segment Table
RE GK1000	
RE TN20	
RE UP421	
RE TH21	Tape Handler
RE TI3274	Tape Image
RE DI3464	Drum Image
RE SD4156	Directory II
RE FD40101	Directory I
RE FA4260	Op File IIa
RE FP7660	Op File IIb
RE OP45215	Op File I
RE LI4160	List I
RE VV2764	Temporaries Phase I
RE TE1300	Temporaries Phase II
RE PP1373	
RE HG674	
RE HH677	
RE HB723	
RE ML747	
RE BB756	
RE BC772	
RE DD1000	
RE BD1003	
RE DA1012	
RE DB1024	
RE DC1035	
RE DF1055	
RE DG1060	
RE DH1070	
RE EE1105	
RE EA1113	
RE EB1120	
RE EC1131	
RE ED1135	
RE FF1145	
RE FB1170	
RE FC1210	
RE MM1223	
RE TT1241	
RE MB1242	
RE CC1260	
RE WM1347	
RE FT1352	
RE HT1360	
RE VC2472	
RE BE76777	
RE EF45213	
RE DE5161	
RE LF3274	

Entrance Phase I

	IA HG	
0	MJ 0 HH	Jump to start
1	MJ 0 BQ6	Alarm exit
2	MJ 0 PP	Jump to Phase II
	CA HG3	
	IA HH	Come from HG
0	RP 15014 HH2	} Clear core from 2764-7777
1	TP CC23 VV	
2	RP 17777 HH4	} Clear drum from 40101-76277
3	TP CC23 40101	
4	RA HH3 MB1	
5	IJ MB HH2	
6	RP 11702 HH10	
7	TP CC23 70076	
10	TP CC25 VV	Set K = 0p
11	TP CC25 VV1	Set b = 0p
12	TV CC31 EB	} Set addresses of first directory
13	TV CC32 EB1	
14	TV CC33 MM11	Set address of List 1
15	TV CC36 HB23	(A) → (A1)
16	TV CC37 BB13	(B) → (B1)
17	TV CC40 EE5	(C) → (C1)
20	TV CC41 EE4	(D) → (C2)
21	TV CC42 EB7	(E) → (E1)

	22	TV CC51 BB12	Enable boxes 3 and 4
	23	MJ 0 HB	
		CA HH24	
		IA HB	Come from HH23
	0	TP CC TH3	} Read 1 block Uniservo 5
	1	RJ TH2 TH	
	2	TP TI24 A	} Identification of Δ FILE (Δ TAPE Δ)
	3	EJ CC11 HB5	
	4	MJ 0 BR12	
	5	RJ TH2 TH	
①	6	TP TI1 A	} Come from HB3 or MB14, read next block
	7	EJ CC4 HB11	
	10	MJ 0 BR12	} Identification of ($\Delta\Delta\Delta$ OP Δ)FILE Δ 1
	11	RJ TH2 TH	
②	12	TP TI1 A	} Read next block
③	13	EJ CC6 HB15	
	14	MJ 0 BB	} Identification of (END Δ OF) Δ ENTRY resp. after change ident. of SUBRTN v changed by MB11 to TT, u changed by MB13
	15	RP 30013 HB17	
	16	TP TI2 5	} Not finished yet, go to ⑤
	17	MJ 10000 HB22	
	20	TP CC23 TN	} Fill constants 5-17 inclusive
	21	MJ 0 HB23	
	22	TP CC57 TN	
④	23	MJ 0 30000	} Do setting for 5 or 7 tapes depend- ing MJ1 test TN = 0 0 0 (5 tapes) TN = 0 3 0 (7 tapes)
		CA HB24	
			Set by HH15 to ML, after all Op Files 1 read in \rightarrow ④ switch

5

IA BB

Come from HB14

- 0 TP CC24 A
- 1 TJ VV2 DD
- 2 EJ VV2 DD
- 3 TP VV2 VV3
- 4 SP VV3 17
- 5 TP A VV5

Is Op File I longer than 1 block?
 equal?
 Smaller?

Store j (length of Op File I item
 within block) in v of VV3 and u
 of VV5

- 6 AT CC47 VV4
- 7 TU VV4 BB10

TI 0 + j → VV4 in u

Set NI

- 10 TP 30000 A
- 11 TP A VV6
- 12 EJ CC12 30000

Place first "CW + # of addresses" of
 next Op File item → VV6

Is there another Op File I following?

- 13 MJ 0 30000

Set by HH22 to BC0, later by MB3 to
 HB11. No.
 Set by HH16 to EE0 later by MB10 to
 ED0. Yes.

B

CA BB14

6

IA BC
0 TP CC15 Q
1 QS VV5 BC4
2 TV VV1 BC5
3 RJ BD4 BD
4 RP 30000 HB11
5 TP TI 30000
CA BC6

Come from BB12
Mask 0 07777 0
Set RP command for bringing Op File
to drum.
Advance and check whether it exceeds
region.
Set by BC1 or DD1
Set by BC2

IA BD
0 RA VV1 VV3
1 QT BC4 Q
2 SP BC5 17
3 SA Q 71
4 TJ CC55 30000
5 TU FT WM
6 MJ O WM
CA BD7

Come from BC3
Advance
CC15 mask is already in Q
Does Op File I exceed region?
Set by RJ in BC3 to BC4
Go to alarm (region exceeded)

	IA DA		Come from MM7, fixed libr. prelim. settings
0	TP CC61 TH3	}	Read 1 block of Tape 1 into buffer
1	RJ TH2 TH		
2	TP CC15 Q	}	Set RP for comparison with List 1
3	QS 5 DB		
4	QT 5 A	}	Set right addr. for first addition to List 1 and index List 1
5	LA A 71		
6	TP A VV14		
7	RA DB2 VV14		Set addr. for next item → List 1
10	RS VV14 CC62		Set index for # of addresses in List 1
11	MJ 0 DC		
	CA DA12		

	IA DB		Come from DG5 with RJ. Fixed libr.
0	RP 20000 DB2	}	Set by DA3 Already in List 1?
1	EJ LI1 DB10		Yes, do nothing.
2	TP A LI		Set by DA7 No., place CW in List 1.
3	RA DB CC53		Adv. u in RP command.
4	RA VV14 CC62		Adv. index for List 1 elements.
5	RA DB2 CC62		Adv. addr. in List 1 for next CW.
6	TJ CC60 DB10	}	Have we exceeded region?
7	MJ 0 MM14		Alarm (too many libr. rout. refer- enced)
10	MJ 0 30000		
	CA DB11		

	IA DC		Come from DA11.
0	TN CC14 Q		Mask 77 77777 0 to Q.
1	QT TI A		CW of Op File 1 fixed libr. → u of A.
2	EJ LI DH		Equal CW in List 1? Yes, go to further handl.
3	TP CC14 Q	}	No
4	TU DC1 DC5		
5	QT 30000 A		
6	LA A 17		
7	AT DC1 DC1		Set next addr. for comparison.
10	TU DC1 DC11	}	
11	TP 30000 A		
12	EJ CC12 DC14		
13	MJ 0 DC		Is next word "74 74747 47474"?
14	RA DC2 CC53		No, go back in loop.
15	TU CC33 DC1		Yes, do settings for next CW in List 1. Reset DC1 to first CW in Op File 1.
16	IJ VV14 DC		Are we thru with List 1? No, go back in loop.
17	MJ 0 MB2		Yes, go to rout. for handl. normal libr. Tape, or changed by ML5 to TT, skip normal libr. Tape.
	CA DC20		

	IA DD	Come from BB1 or BB2
0	RS VV2 CC24	Form words in Op File 1 minus 170
1	TU CC36 BC4	Set RP command BC4
2	MJ 0 BC2	
	CA DD3	
	IA DF	Come from DH14 or DG7, fixed libr.
0	RJ EB7 EB	Place CW and drum addr. → Direct 1 and adv. counters.
1	TP CC14 Q	Restore mask in Q.
2	MJ 0 DC14	Back to handl. next List 1 word.
	CA DF3	
	IA DG	Come from DH13, fixed libr.
0	TU DC1 DG4	Set addr. of Op File CW in DG4 (after equality).
1	RA DG4 CC53	Adv. by 1.
2	RA DG4 CC53	Adv. by 1.
3	TP CC13 Q	} Put cross reference → A.
4	QT 30000 A	
5	RJ DB10 DB	Go to handl. cross reference.
6	IJ VV15 DG2	Are all cross ref. handled? No, back in loop.
7	MJ 0 DF	Yes, go to handl. CW.
	CA DG10	

	IA DH	
0	TU DC1 DH11	Set RP command.
1	TU DC1 DH3	Set addr. DH3.
2	TP CC14 Q	Mask
3	TP 30000 VV6	Save CW + # of lines in Op File item in VV6.
4	QT VV6 VV15	Save # of lines in Op File item in VV15.
5	TV VV1 DH11	Place next drum addr. for Op File item in v of DH11
6	SP VV15 17	Set addr. DH7
7	AT CC63 DH10	
10	0 0 0	Place Op File item → drum.
11	TP 30000 30000	
12	RA VV1 VV15	Adv. drum addr.
13	RS VV15 CC64	Subtr. 3 from # of lines to get index for cross ref.
14	SJ DF DG	Ind. neg; skip hdl. cross ref., pos. go to hdl. cross ref.

CA DH15

(C1)

	IA EA	Come from EE5
0	SP VV 17	K → 1st word Direct. II drum addr. of first statem. CW in Op File I → u of SD
1	TU A SD	
2	TV CC41 EE5	(C) → (C2)
3	TV CC45 EE4	(D) → (D2)
4	MJ 0 EB	→ (C2)
	CA EA5	

	IA EB		Come from EA4 or EE4 or EE5
(C2)	0 TU VV6 30000		Set by HH12 to [FD1] CW → FD1
	1 TV VV 30000		Set by HH13 to [FD2] K → FD2
	2 RA FD CC22		Adv. counter in FD by 2 in u
	3 TV VV6 VV2		v of j'th word → temp 1
	4 RA VV VV2		K + temp 1 → K. Updates drum addr.
	5 RP 20002 EB7	}	next item.
	6 RA EB CC21		Adv. addr. in Dir. I for next CW.
(7)	7 TJ CC56 30000	}	Set by HH21 to FB, later by MB12 to
	10 MJ 0 BD5		FF, in between used by DF5 in RJ.
	CA EB11		Is Directory 1 too long on drum? Jump to alarm: THE PROBLEM IS TOO LONG.

	IA EC		Come from EE4
(D2)	0 SP VV 17	}	
	1 TU A SD1		K → 2nd word Directory II
	2 TV CC41 EE4		(D) → (C2) set in order to skip this
	3 MJ 0 EB		region later
	CA EC4		→ (C2)

(B2) 0 IA ED
 1 TP CC15 A }
 2 QS 5 ED4 }
 3 TU VV6 VV7 }
 4 TP VV7 A }
 5 RP 20000 ED6 }
 6 EJ LI EE }
 7 TV VV6 VV2
 MJ 0 FC11
 CA ED10

Come from BB13

Fill u of RP

CW → A

Is CW in List 1?

v of j'th word → temp 1

→ (10)

(B1) 0 IA EE
 1 TP CC16 Q }
 2 QT VV6 A }
 3 EJ CC17 EE5 }
 4 EJ CC20 EE5 }
 (D) 5 MJ 0 30000
 (C) MJ 0 30000
 CA EE6

Come from BB13

Is first CW in Op File item 26--- Or 27---?

Set by HH20 to EB, later by EA3 to EC and by EC2 to EB
 Set by HH20 to EA, later by EA4 to EB

(E1)

IA FB
 0 TP VV2 A }
 1 AT VV3 VV10 }
 2 TP CC24 A }
 3 TJ VV10 FB6 }
 4 EJ VV10 FB6 }
 5 MJ O FC11
 6 TV VV1 FB10 }
 7 RP 30170 FB11 }
 10 TP TI 30000 }
 11 RA VV1 CC24
 12 TP VV2 A }
 13 AT VV3 A }
 14 ST CC24 VV2 }
 15 TP CC23 VV3
 16 RJ TH2 TH
 17 MJ O FB
 CA FB20

Come from EB7
 Temp 1 + j → VV10
 Temp 1 + j ≥ 120 → FB6
 No → (10)
 Case when file exceeds block
 TI → drum
 G + 120 → G
 Temp 1 - (120-j) → temp 1
 j = 0
 Bring next block to core
 → (E1)

9

IA FC
 0 SP VV2 17
 1 TP A VV13
 2 TP CC15 Q
 3 QS VV13 FC6
 4 TV VV1 FC7
 5 TU VV4 FC7
 6 RP 30000 FC10
 7 TP 30000 30000

Come from FF5

RP 3 (temp 1)
 TP TI + j [drum addr.]

Bring Op File I item to drum

10

10 RA VV1 VV2
 11 RA VV2 VV3
 12 MJ 0 HB12
 CA FC13

Come from FC10 or FB5 or ED7

→ 3

(E2)

IA FF

0 TP VV2 A }
 1 AT VV3 VV10 }
 2 TP CC24 A }
 3 TJ VV10 FF6 }
 4 EJ VV10 FF6 }
 5 MJ O FC }
 6 TP CC50 A }
 7 ST VV5 VV11 }
 10 TP CC15 Q }
 11 QS VV11 FF14 }
 12 TU VV4 FF15 }
 13 TV VV1 FF15 }
 14 RP 30000 FF16 }
 15 TP 30000 30000 }
 16 RA VV1 VV11
 17 RS VV2 VV11
 20 TP CC23 VV3
 21 RJ TH2 TH
 22 MJ O FF
 CA FF23

Come from EB7

Temp 1 + j → VV10

Temp 1 + j ≥ 120 → FF6

No → (9)

Case file exceeds block

Library Op File → drum

G + (120-j) → G

Temp 1 - (120-j) → temp 1

j = 0

→ (E2)

7

IA MB

0 0 0 2 }
 1 0 0 7777 }

2 TP CC26 TH3 }
 3 TV BC4 BB12 }
 4 RJ TH2 TH }

5 TP TII A

6 EJ CC4 MB10

7 MJ 0 BR7

10 TV CC44 BB13

11 TV CC43 HB13

12 TV CC46 EB7

13 RA HB13 CC22

14 RP 10002 HB11 }
 15 TP CC23 VV2 }

CA MB16

Come from DC16 after List 1 is read to drum

Gets changed constants used by HH for presetting used by HH4

Read 1 block of Tape II and disable boxes 3 and 4

Put second word → A

Is it "FILE Δ 1"?

No; alarm.

ⓑ → ⓑ2
 ⓐ → ⓐ2
 ⓔ → ⓔ2

Change test for "end of entry" "to" "libr. subrout"
 → ①

Clear VV2 words in block and VV3 words in Op File 1 item j

IA ML

0 TP 5 A }
 1 EJ VC10 VC23 }

2 SJ ML3 MM

3 TJ CC26 ML5

4 MJ 0 MM1

5 TV CC43 DC17 }
 6 MJ 0 MM1 }

CA ML7

Come from HB23

Case no library rout at all skip handling libraries but jump to adv. tape 1 by 1 block VC23 is patch in constant pool FC23 of Phase II

Only variable library

Only fixed library

Both libraries

Only fixed library

	IA MM		Come from HB23
	0 TV CC66 DA11		Entry for only variable library, change exit
(A1)	1 RJ TH2 TH		Read next block on Tape 5
	2 TP TI A	}	
	3 EJ CC7 MM5		Is first word List 1?
	4 MJ O BR12		Jump to alarm: LIST 1 LABEL IN- CORRECT.
(4)	5 RJ TH2 TH		Read next block of Tape 5
	6 TP TI1 A	}	Is it END OF ENTRY?
	7 EJ CC6 DA		Exit to DA handl. of fixed libr.
	10 RP 30170 MM12	}	
	11 TP TI 30000		Set by HH14 to LI. Read whole List 1 to LI on drum
	12 RA MM11 CC24		
	13 TJ CC65 MM5		Does it exceed region?
	14 TU HT WM	}	
	15 MJ O WM		Go to alarm: TOO MANY LIBR. ROUT. REFERENCED.
	CA MM16		
	IA TT		Come from ML1 (with no List 1) rsp HB13 with List 1
(A2)	MJ O HG2		Exit out of Phase I, go to Phase II
	CA TT1		

Alarm Routine

IA WM

0 TP 30000 UP3

1 RJ UP2 UP

2 MJ 0 HG1

CA WM3

IA FT

0 0 FT1 0

1 0 FT2 4 Alarm, the problem is too long.

2 66 33300 15254

3 51 25463 04701

4 34 65016 65151

5 01 46515 03222

CA FT6

IA HT

0 0 HT1 0 Alarm, too many library routines
referenced in the problem.

1 0 HT2 11

2 66 51510 14724

3 50 73014 63425

4 54 24547 30154

5 51 67663 45030

6 65 01543 03130

7 54 30502 63027

10 01 34500 16633

11 30 01525 45125

12 46 30472 27777

CA HT13

Constants

	IA	CC
0	50	00105 TI
1	01	01323 05001
2	01	66245 23001
3	01	01015 15201
4	31	34463 00104
5	30	50270 15131
6	01	30506 65473
7	46	34656 60104
10	65	67255 46650
11	01	31344 63001
12	74	74747 47474
13	0	77777 0
14	0 0	77777
15	0	7777 0
16	0	77000 0
17	0	27000 0
20	0	26000 0
21	0 0	2
22	0 2	0
23	0 0	0
24	0 0	170
25	0 0	0P
26	50	00102 TI
27	10	5 0

30 10 2 0
31 0 0 FD1
32 0 0 FD2
33 0 TI LI
34 0 0 6
35 0 0 7
36 0 30170 ML
37 0 0 EE
40 0 0 EA
41 0 0 EB
42 0 0 FB
43 0 0 TT
44 0 0 ED
45 0 0 EC
46 0 0 FF
47 0 TI 0
50 0 170 0
51 0 0 BC
52 0 0 77
53 0 1 0
54 0 0 5
55 TP TI BE
56 TV VV EF
57 0 3 0
60 TP A DE
61 50 101 TI

62 0 0 1
63 RP 30000 DH12
64 0 0 3
65 TP TI DE
66 0 0 MB2
CA CC67

Temporaries VV 2764 Segment. Phase I

VVO	()		K drum address of beginning of item
1	()		G drum address next TI item on drum
2	()		Temp 1 # of word handled already within this block
3	()		j (in v) length of Op File to be handled next
4	()		TI 0 + j (in u) addr. in tape image where next Op File starts
5	()		j (in u)
6	()	()	Current word
7	()		Call word (in u)
10			Temp 1 + j
11			120 - j
12			Temp 2 = 0 (temp 1) 0
13			Temp period
14			Ind. List 1 and working space
15			Ind. for cross ref. in Op File 1 item of fixed libr. and working space

Segmentation Phase II

RE VV2764	Temporaries Phase I
RE MB1242	
RE CF2627	
RE LM2641	
RE FA4260	Op File IIa
RE ZA77000	
RE BQ632	
RE BR537	
RE ST653	Segment table
RE GK1000	
RE TN20	
RE UP421	
RE TH21	Tape Handler
RE TI3274	Tape Image
RE DI3464	Drum Image
RE SD4156	Directory II
RE FD 40101	Directory I
RE FP 7660	Op File IIb
RE OP45215	Op File I
RE LI4160	List I
RE TE1300	Temporaries Phase II

RE BB1373
RE CC1520
RE CD1536
RE CE1567
RE PT1600
RE DD1602
RE EE1617
RE EF1657
RE EG1672
RE EH1724
RE EI1754
RE EJ1766
RE EK2001
RE EL2004
RE EM2017
RE EN2031
RE E02064
RE EP2067
RE EQ2106
RE ER2117
RE ES2124
RE ET2131
RE EV2135
RE EW2150
RE EX2153
RE EY2156

RE EZ2163
RE LL2165
RE FF2173
RE FG2266
RE GG2273
RE GH2325
RE GI2360
RE GJ2404
RE GL2407
RE HH2414
RE RC2464
RE FC2472
RE WN2567
RE BU2572
RE EU2606
RE FU2614
RE FB7660
RE DA1012
RE TT1241

RE WSO
RE S010000
RE SM7660
RE SN10000

	IA BB		Come from HG2 Phase I
0	MJ 0 BB3		Jump to start
1	MJ 0 BQ6		Alarm exit
2	MJ 0 ZA10		Normal exit
3	TP FC12 Q	}	
4	QT 10 TE26		
5	TP FC63 A		
6	ST TE26 A		
7	TV A 10		
10	MJ 0 BB14		
11	QT 7 TE26		
12	EJ FC7 BB123	}	Is $(7)_v$ zero? Yes, clear 7
13	MJ 0 BB20		
14	QT 10 A	}	
15	SS 7 17		
16	TU A 7		
17	MJ 0 GI		
20	TU FD BB73	}	Go to form segm. length for this probl.
21	RA BB73 FC10		
22	TU SD BB42		Set MD address of first statement
23	RP 10005 BB25	}	
24	TP FC7 TE22		
25	RP 10021 BB27		
26	TP FC7 TE		
27	TP FC17 TE5	}	Clear temporaries
30	TP FC17 TE6		
			Set α and β to same fixed Op File IIa addr.

①

31	TP FC20 TE7		Set γ to fixed Op File IIb addr.
32	TP FC7 FP		Clear first word of Op File IIb area
33	RP 13400 BB35	}	Clear Op File IIa area
34	TP FC7 FA		
35	TU BB42 BB37		Come from BB34 or EE34
36	TP FC45 A	}	Set u of repeat command BB41
37	SA 30000 17		
40	TU A BB41		
41	RP 30000 BB43	}	Transfer next statement Op File item to drum
42	TP 30000 DI		
43	TP DI TE10		Record 1st word of statement Op File in Temp 1
44	TV DI1 TE1	}	# of lines in this statement rout. → t and t_2
45	TV DI1 TE4		
46	TV TE5 BB47		
47	TU TE10 30000		α address to NI
50	TV TE5 BB52		Statement call word → Op File IIa
51	RA BB52 FC5		α address to BB52
52	TV DI1 30000		$\alpha + 1$
53	RJ CE1 CE	}	# of lines in routine → $\alpha + 1$
54	RA TE2 FC13		
55	MJ 0 BB56		Advance α and K by 2 and check exceeded region
56	TP FC3 Q	}	Free but needed
57	QT TE10 A		
60	EJ FC46 DD		

4

61 TV DI TE15
 62 RS TE15 FC6
 63 TV FC47 TE11
 64 IJ TE15 CC
 65 MJ 0 BB66

Come from BB60 or BB120
 Set index for # of cross ref. (one higher since ind. jump in beginning
 Set addr. of first cross ref → TE11
 Are all cross references handled?
 Free but needed

6

66 RA TE6 FC6
 67 EJ TE5 EE
 70 SP A 17
 71 TU A BB72
 72 SP 30000 0
 73 RP 30000 BB75
 74 EJ FD1 BB77
 75 MJ 0 BR5

Adv β by 2
 $\beta = \alpha$? yes, go to switch
 Do handl. of β 's
 Next CW to be processed for $\beta \rightarrow A$
 Set by BB71
 Set by BB20, 21
 Search Directory I for call word at addr. β
 Go to alarm 5 (Directory incorrect)

76 02 0 0
 77 SN Q 17
 100 SA BB73 0
 101 SA BB74 0
 102 TU A BB103
 103 SP 30000 17
 104 TU A BB112
 105 TU BB112 BB107
 106 TP FC45 A
 107 SA 30000 17
 110 TU A BB111
 111 RP 30000 BB113
 112 TP 30000 DI

$-j-n+r$ in u
 $(-j-n+r)+(j+n) = r$
 Set u of BB103 $r + FD1$
 Set by BB102
 Set this Directory I addr. into u of BB112
 30000 in v of A
 Set u address of BB112
 Set by BB105
 Set by BB110 transfer cross ref.
 Op File I item
 Set by BB104 from MD to drum image

113	RA TE1 DI1		Add # of lines in this routine to t
114	TV TE6 BB116	}	
115	RA BB116 FC5		Enter # of lines in this routine in the address following the CW in Op File IIa
116	TP DI1 30000		
117	SP DI 0	}	Is 77000 > CW? If no, then 77__
120	TJ FC3 BB61		→ (4) go to set index for handl. cross ref. Go to handle 77__ case
121	RF CF11 CF		
122	MJ 0 BB66		→ (6) (has no cross ref., therefore no index needed)
123	TP FC7 7	}	Come from BB13
124	MJ 0 BB20		Setting of 00007 in case no single valued variables
	CA BB125		

	IA CC		Come from BB64
0	SP TE11 17	}	
1	TU A CC2		Put CW at addr. given by TE11 in NI
2	TP 30000 TE20		Save found CW in TE20
3	TP FC3 Q	}	Mask
4	QT TE20 A		
5	EJ FC53 CC14		27---?
6	EJ FC46 CC14		26---?
7	TP FC54 Q	}	
10	QT TE20 A		4----?
11	EJ FC55 HH		
12	RJ CD13 CD		None of the three cases for Op File IIa
13	MJ 0 BB64	}	26--- or 27---
14	RJ CD30 CD14		Case for Op File IIb
15	MJ 0 BB64		
	CA CC16		

	IA CD	Come from CC12 or . . .	} Op File IIa
0	TU TE2 CD3	Set n in u of RP command CD3	
1	RA CD3 FC10		
2	SP TE20 0	Check whether CW already in Op File IIa	
3	RP 30000 CD5		
4	EJ FA2 CD11		
5	TV TE5 CD6	Set v of NI	
6	TP A 30000	Place CW in Op File IIa	
7	RJ CE1 CE	Adv. by 2 in v and check over- flow and take care of 77 case	
10	RA TE2 FC13	Adv. by 2 in u	
11	RJ CD11 CD12	Exit possible for Vary (DD region)	
12	RA TE11 FC5	Adv. to next cross ref. address	
13	MJ 0 30000	Exit for RJ	
14	SP TE7 0	Come from E01 or EP5 or . . .	
15	SS FC20 17	Next addr. in Op File IIb Subtract first addr. and shift to u	
16	TU A CD21	Set n in u of RP command CD21	
17	RA CD21 FC10	Check whether CW already in Op File IIb	
20	SP TE20 0		
21	RP 30000 CD23		
22	EJ FP1 CD26		
23	TV TE7 CD24	Set v of NI	
24	TP A 30000	Place CW in Op File IIb and adv.	
25	RJ CE5 CE4		

26	RJ CD26 CD27	Exit possible for Vary
27	RA TE11 FC5	Adv. to next cross ref. address
30	MJ 0 30000	Exit of RJ
	CA CD31	

Op File
IIB

	IA CE				
0	RA TE5 FC6	}		Come from CF or CF2	
1	TJ FC64 30000			Adv. next addr. for Op File IIa by 2 and check whether region exceeded	Op F IIa FA = 4260
2	MJ 0 EW				FC64 = SM = 7660 space for 3400
3	0 0 0		Free		Op F IIb FP = 7660
4	RA TE7 FC5	}		Come from CD25	FC65 = SN
5	TJ FC65 30000			Adv. next addr. for Op File IIb by 1 and check whether region exceeded	= 10000 space for 120
6	MJ 0 EW				
7	0 0 0		Free		
	CA CE10				

	IA	PT		(patch)
0	TJ	TE1	EH10	Come from EH3
1	MJ	0	EQ2	
	CA	PT2		

	IA CF		Come from BB121
0	RA TE12 DI1		Adv. TE12
1	SP MB1 1	}	Is # of elements > 17776?
2	TJ DI1 CF6		
3	SP MB1 0	}	Is # of elements > 7777?
4	TJ DI1 CF7		
5	MJ 0 CF10	}	Adv. TE22
6	RA TE22 FC5		
7	RA TE22 FC5		
10	RA TE22 FC5		
11	MJ 0 30000		Exit used only in RJ, by BB121
	CA CF12		

	IA DD		Come from BB57 after detected 26---CW
0	TP FC12 Q	}	Mask out # of lines
1	QT TE10 A		
2	EJ FC57 DD5		Is it 4 ? Or 5 ? Go NI
3	TP DI4 TE20	}	Store (after check) library rout. in Op File IIa
4	RJ CD11 CD		
5	TP DI3 TE20	}	Store (after check) last (jump out) CW in Op File IIb
6	RJ CD26 CD14		
7	TV FC61 EE		Set switch
10	RJ DD10 DD12		Chance for jumping once in Vary start (orig. set to jump, reset by EF14)
11	MJ 0 BB66		Exit to (6)
12	TP DI2 TE31	}	Save CW of last statem in Vary loop Save addr. in Op F IIa of first statem CW in Vary loop
13	TV BB47 TE23		
14	MJ 0 DD11		
	CA DD15		Done once in Vary loop

	IA EE		Come from BB67
0	MJ 0 EE1		Switch for Vary set by DD7 to EF (Vary), restored by EE30
1	RA TE TE1		Come from EE or EF11 form $T + t \rightarrow T$
2	SP TE 0	}	is $M > T$?
3	TJ TE21 EE35		
4	RS TE TE1		$T - t \rightarrow T$
5	ZJ EE6 EG2	}	Is $T = 0$? (i.e. are we at beginning of segment?)
6	TP TE21 A		
7	TJ TE1 EE14		
10	SP TE13 17	}	Case $t < M$ and next T would be $> M$ TE13 was filled when bef. at EE3 we had gone to EE17
11	TU A EE12		
12	TP 30000 FB		
13	MJ 0 EI		Go and make segment with settings before
14	TP FC61 A	}	Come from EE7 are we in Vary loop? Yes; go to handl. Vary case No; go to alarm (statem. too big)
15	EJ EE EK		
16	MJ 0 EG5		
17	TV FC62 EE	}	Come from EE3, case $M > T$ Restore switch to Non-Vary case Reset RJ to beginning position for Vary loop
20	TV FC50 DD10		
21	TP TE5 TE13	}	Save values of this case, for the case we overshoot with next statem.
22	TP TE7 TE14		
23	TP TE12 FA1		
24	RA TE3 TE4		

$\alpha \rightarrow$ Temp 4 = TE13
 $\gamma \rightarrow$ Temp 5 = TE14
 TE12 = Temp 3 \rightarrow
 second word in
 Op File IIa
 $T2 + t2 \rightarrow T2$

25	TP FC12 Q	}	Come from EF6 Case we are not yet at end of Vary loop <u>or</u> segm. length, Mask 0 0 77777 → Q Adv. u portion of transfer command by # words in Op File item Set u in BB42
26	QT TE10 A		
27	SP A 17		
30	AT BB42 BB42		
31	TP FC11 Q	}	Have all statem. CW's been processed? i.e., entered in Op File IIa? Detects end of needed information
32	QT A A		
33	EJ SD1 FF		
34	MJ 0 BB35		No, go back to handle next CW
35	TP FC7 TE30	}	Patch, clear TE30 and save TE22 in TE40
36	TP TE22 TE40		
37	MJ 0 EE17		
	CA EE40		

	IA EF		Case we are inside Vary loop Come from EEO after having been in DD
0	RA TE24 TE1		$t_4 + t = t_4$
1	RA TE25 TE4		$t_5 + t_2 = t_5$
2	TP FC11 Q	}	Mask 0 77777 0 \rightarrow Q
3	QT TE31 TE26		
4	QT TE10 A		Are we at end of Vary loop?
5	EJ TE26 EF7		Yes
6	MJ 0 EE25		No (skip saving α and γ since in Vary only first import.)
7	TP TE24 TE1		$t_4 \rightarrow t$
10	TP TE25 TE4		$t_5 \rightarrow t_2$
11	RP 10002 EE1	}	
12	TP FC7 TE24		Clear t_4 and t_5
	CA EF13		

	IA EG		Case $T > M$, yet we are at the beginning of segment (implies $t > M$)
0	RJ EG EG1	}	Come from E01. RJ exit for EP15 Patch to EQ. Exit after region check when region not exceeded
1	MJ O EN31		
2	TP FC3 Q	}	Come from EE5 or EW2
3	QT FA2 A		
4	EJ FC46 EJ		
5	TU BU WN	}	Is CW at beginning of Op File IIa a 26---?
6	MJ O WN		
7	TP FC7 TE27	}	Go to alarm (segm. too big)
10	TP FC7 TE30		
11	RJ EG11 EG12		Clear t_1 and t_3
12	TV TE5 EG14	}	Inserted for jump out of EK
13	RA EG14 FC5		
14	RA TE27 30000		
15	TV EG14 EG16	}	Come from EG11 or EH11 or EG27
16	RA TE 30000		
17	TP TE21 A	}	$t_1 + (\alpha + 1) \rightarrow t_2$
20	TJ TE EH12		
21	MJ O EX		$T + (\alpha + 1) \rightarrow T$
22	LA A 17	}	Is $T > M$? \rightarrow (37)
23	TU A EG25		
24	TP FC3 Q		
25	QT 30000 A		
26	EJ FC46 EH		Adv. α by 2 taken out here, see EG30 now
27	MJ O EG12		Is CW at α a 26---?

30 RA TE5 FC6 }
31 MJ 0 EG22 }
CA EG32

Patch, come from EM7 adv. α by 2
(Patch for replaced instruct. EG21)

	IA EH		Come from EE15 or EG26. Case $t > M$ and we are not at beg. of segm.
0	RA TE30 FC5		Set $t_3 = 1$. Index for Vary within Vary
1	RS TE1 TE27		Come from EK2 or EH $t - t_1 = t$
2	TP TE21 A	}	Is $t > M$?
3	TJ TE1 EH10		
4	SP TE5 17	}	Case $t < M$
5	TU A EH6		Record statem. CW into first CW of Op File IIB (for IP command.)
6	TP 30000 FP		
7	MJ 0 EI2		Go to D, exit to write on tape
10	RA TE1 TE27		$t + t_1 = t$ restore t since it is still too big and we have to make beginning segment bigger
11	MJ 0 EG12		Go to (34)
12	TV TE5 EH14	}	Come from EG20 or EH27
13	RA EH14 FC5		We finally made first part too big and have to go back to last statement and form segm. resp. back to last Vary within Vary
14	RS TE 30000		
15	SP TE5 17	}	CW at $\alpha = 26$ ---
16	TU A EH20		
17	TP FC3 Q		
20	QT 30000 A		
21	EJ FC46 EQ		
22	EJ FC53 EH24		CW at $\alpha = 27$ ---? Statem. CW, go to check whether Vary inside Vary

	IA EJ		Come from EG4 resetting to Vary in beginning of segm.
0	TP FC17 TE5	}	
1	TP TE7 TE32		Save TE7= γ in TE32 and set counters back to segment start
2	TP FC20 TE7		
3	TP FC7 TE12		
4	MJ 0 EJ11		Jump to clearing TE22, t_1 , t_3 and exit
5	TP TE13 TE5	}	Come from EK
6	TP TE7 TE32		Save TE7 = γ in TE32 and set counters back to Vary start
7	TP TE14 TE7		
10	TP FA1 TE12		
11	TP FC7 TE22		Clear TE22
12	MJ 0 EG7		Jump to clearing t_1 and t_3 and to exit from RJ
	CA EJ13		
	IA EK		Come from EE15 resetting to Vary not in beginning of segment.
0	RJ EG11 EJ5		Do resetting of counters
1	TP TE40 TE22		Reset TE22
2	MJ 0 EH1		
	CA EK3		

	IA EL		Come from LL1 or LL3
0	TP FC3 Q	}	
1	SP TE5 17		
2	TU A EL3		Is CW at hand a 77---?
3	QT 30000 A		
4	EJ FC3 EL6		
5	MJ 0 30000		Not 77---, skip changing TE12 and TE22 (set to ER or to EL12) by EM by EM3
6	0 0 0	}	Adv. resp. reduce TE22
7	TV TE5 EL11		Adv. resp. reduce TE12 by # of instructions
10	RA EL11 FC5		
11	0 0 0	}	Used !!!
12	0 0 0		Jump back to rout.
	CA EL13		

	IA EM		Come from EX1
0	TV FC74 EL5	}	Set "ER" in v of EL5
1	RP 30002 LL		Entrance for going forward
2	TP EM6 EL11		
3	TV ER2 EL5	}	Come from EH23 Set "EL12" in v of EL5
4	RP 30002 LL2		Entrance for going backward
5	TP EM10 EL11		
6	RA TE12 30000	}	Const. for EL11, EL12 in forward case
7	MJ 0 EG30		
10	RS TE12 30000	}	Const. for EL11, EL12 in backward case
11	MJ 0 EH26		
	CA EM12		
	IA LL		
0	TP LL4 EL6		Come from EM1
1	MJ 0 EL		Forward
2	TP LL5 EL6		Come from EM4
3	MJ 0 EL		Backward
4	RJ LM14 LM		Const. for forward
5	RJ LM20 LM15		Const. for backward
	CA LL6		

	IA LM		Come from EL6
0	TP FC5 A	}	Entrance for forward
1	SA TE5 17		
2	TU A LM3		
3	TP 30000 TE34		
4	SP MB1 1		
5	TJ TE34 LM11	}	Is # of inst. > 17776?
6	SP MB1 0		
7	TJ TE34 LM12	}	Is # of inst. > 7777?
10	MJ 0 LM13		
11	RA TE22 FC5		
12	RA TE22 FC5	}	Adv. resp. reduce TE22
13	RA TE22 FC5		
14	MJ 0 30000		Exit, used in RJ from EL6 for forward case
15	RP 20003 LM17	}	Entrance for backward
16	RA LM11 BB76		
17	RJ LM14 LM		Change RA to RS in LM11-13
			Handle TE22 for 77 ... when going backward
20	RP 20003 30000	}	Exit, used in RJ from EL6 for backward case
21	RS LM11 BB76		
	CA LM22		

	IA EN		Come from EQ10
0	TP TE16 A	}	Are we finished with all CW's in Op File IIa?
1	EJ TE5 EH4		
2	SP TE16 17	}	Exit
3	TU A EN5		
4	TP BB73 EN6		Search Dir. I for CW given at new α
5	SP 30000 0		Set by EN3
6	0 0 0		Set by EN4. RP ... BB75 exit to alarm
7	EJ FD1 EN10		
10	SN Q 17		}
11	SA EN6 0		
12	SA EN7 0		
13	TU A EN14		
14	SP 30000 17	}	Set by EN13
15	RJ EN15 EN16		Set u of EN24. RJ for use in addr. ET1
16	TU A EN24		
17	TU EN24 EN21	}	Set u of EN23
20	TP FC45 A		
21	SA 30000 17		
22	TU A EN23	}	Set by EN22 Transfer Op File I item to drum image Set by EN16
23	RP 30000 EN25		
24	TP 30000 DI		

25	TP FC3 Q	}	
26	QT DI A		Is CW 26---?
27	EJ FC46 EO		Go to handle I Ib 26---
30	EJ FC53 EP		Is CW 27---? Go to handle I Ib 27---
31	RA TE16 FC6		Come from EN30 or EG1 or EP2. Adv. new α addr. by 2
32	MJ 0 EN		Go back in loop
	CA EN33		
	IA EO		Come from EN27 CW 26---
0	RA TE7 FC5		Adv. Op File I Ib address
1	TJ FC65 EG		Did we exceed region? No, go via patch EG-EG1 to EN31
2	MJ 0 EY		Yes; jump to make segment
	CA E03		

	IA EP		Come from EN30	CW27---
0	TP FC12 Q	}	Mask out # of lines	
1	QT DI A			
2	ST FC6 TE17		Set index for # of cross ref. (1 too high, pre-done)	
3	TU DD12 EP6		Preset EP6 to DI2	
4	IJ TE17 EP6		Are all done?	
5	MJ O EN31		Yes	
6	TP 30000 TE20		(Starts with DI2)	
7	TP FC3 Q	}	Mask out CW code	
10	QT TE20 A			
11	EJ FC46 EP15		CW = 26---?	
12	EJ FC53 EP15		CW = 27---?	
13	RA EP6 FC51		Adv. addr of cross ref.	
14	MJ O EP4		Go to handle next cross ref.	
15	RJ EG EO	}	Put CW in Op File IIb resp. adv., Op File IIb addr. (since the CW is already in from the first try)	
16	MJ O EP13			
	CA EP17			

	IA EQ		Come from EH21
0	RS TE30 FC5		Subtract indicator for Vary in Vary by 1
1	RJ EQ1 EQ2		Inserted for RJ use by EY
2	TP FC20 A	}	Has anything been in Op File I Ib before we went back? No, skip the part EQ (TE32 set by EJ1 or EJ6)
3	EJ TE32 EH4		
4	TJ TE7 EQ7		Is $\gamma > FC20$? (0 0 FP1)
5	TP FC17 TE16	}	Case T was = 0 } Set new α in either case to starting addr. Case T was \neq 0 }
6	MJ 0 EQ10		
7	TP TE13 TE16		
10	MJ 0 EN		Go to handle case Op File I Ib
	CA EQ11		
	IA ER		Come from EL5. Handle TE3 for going forward
0	EJ FC46 ER3	}	Is CW 26---? Is CW 27---?
1	EJ FC53 ER3		
2	MJ 0 EL12		Skip changing TE3 = T2 (v used as constant also; by EM3)
3	TU FC74 EL11		Change in EL11 the TE12 \rightarrow TE3
4	MJ 0 EL7		Jump to change TE3 (but not TE22!)
	CA ER5		
	IA ES		Come from EH25
0	TV TE5 ES2	}	Handle TE3 for going backward Inserted for use in RJ by EV
1	RA ES2 FC5		
2	RS TE3 30000		
3	RJ ES3 ES4		
4	MJ 0 EH26		
	CA ES5		

	IA ET		Come from ET3 or GI14
0	SP TE13 17	}	
1	RJ EN15 EN3		Set addr. BB42
2	TU A BB42		
3	MJ O 30000		
	CA ET4		
	IA EV		Come from EH27
0	SP TE5 17	}	
1	TU A EV3		
2	TP FC3 Q		CW at $\alpha = 26$ ----
3	QT 30000 A		
4	EJ FC46 EV11		
5	EJ FC53 EV7		CW at $\alpha = 27$ ----
6	MJ O EH12		Exit after handling either case
7	TP TE30 A	}	Is TE30 = 0?
10	ZJ EV6 EV11		
11	RJ ES3 ES	}	
12	MJ O EV6		
	CA EV13		
	IA EW		Come from CE2 or CE6 (Op File IIa or b exceeded)
0	TP FC7 A	}	Is T = 0?
1	TJ TE EE10		
2	MJ O EG2		Yes; start normal Vary way for T = 0
	CA EW3		

	IA EX		Come from EG21
0	TP TE5 A	}	Is Op File IIa exceeded?
1	TJ FC64 EM		No
2	MJ O EH12		Yes; go backward where segm. can be made
	CA EX3		
	IA EY		Come from EO2 (after Op File IIb region exceeded)
0	RS TE5 FC6		Reduce α by 2
1	RJ EQ1 EH12		Go back to next possible break off point
2	TP TE5 A	}	Is α farther back than TE16?
3	TJ TE16 EZ		Yes; break off
4	MJ O EY1		No; go back in loop
	CA EY5		
	IA EZ		Come from EY3
0	TP FC20 TE7	}	Do setting for final Op File IIb (after it was exceeded) and jump to handling Op File IIb
1	MJ O EQ5		
	CA EZ2		

	IA FF		Come from EE33 or
0	RJ GJ2 GJ	} Done once	} Jump first to set TE41, later set by FF1 to MJ 0 FF12 Set switch in FF to MJ 0 FF12 Set Z's in beginning of first block Set FILE Δ Δ TWO Δ Δ Δ done once Fill up with Z's Set tape hdl. code for right tape (3 or 6) Write first block
1	TP RC1 FF		
2	RP 10024 FF4		
3	TP FC35 TI		
4	TP FC27 TI24		
5	TP FC30 TI25		
6	RP 10142 FF10		
7	TP FC35 TI26		
10	RJ FG2 FG		
11	RJ TH2 TH		
12	TP FC31 TI		
13	TP FC32 TI1		
14	SP TE13 0		
15	ST FC17 TI2		
16	TP FA1 TI3		
17	SP TE 0		
20	SS FA1 0		
21	AT FC6 TI4		
22	SP TE41 0	} Address for IP command (N + T ₂ + 1)	
23	AT TE3 TI5		
24	RP 10162 FF71	} Fill first block up with Z's	
25	TP FC35 TI6		
26	RJ TH2 TH		Write first block on tape 3

27	SP TI2 0	}	Set index how many more
30	DV FC21 TE16		
31	TP A TE17	}	Remainder saved for # of words left over
32	TP RC2 FF70		
33	TU FC26 FF37	}	Set beginning addr. FA2 (Op File IIa) in FF37 Are all complete blocks written?
34	IJ TE16 FF36		
35	MJ 0 FF43	}	Yes; go to handle last fractional block Come from FF34
36	RP 30170 FF40		
37	TP 30000 TI	}	[FA2] in beginning set by FF33
40	RJ TH2 TH		
41	RA FF37 FC22	}	Write next block Adv. addr. Op File II
42	MJ 0 FF34		
43	SP TE17 0	}	Go back in loop for next block Come from FF35 after all whole blocks written. Are there words for partial block?
44	ZJ FF45 FF63		
45	SP A 17	}	Set last words in partial block
46	AT RC4 FF50		
47	TU FF37 FF51	}	Set last words in partial block
50	RP 30000 FF52		
51	TP 30000 TI	}	Fill rest of block with Z's
52	SP TE17 17		
53	SS FC22 0	}	Fill rest of block with Z's
54	SN A 0		
55	AT RC5 FF57	}	Fill rest of block with Z's
56	RA FF60 TE17		
57	RP 30000 FF61	}	Fill rest of block with Z's
60	TP FC35 TI		

61	RS FF60 TE17		Restore last instr.
62	RJ TH2 TH		Write last data block
63	TP FC36 TI	}	Place end of entry
64	TP FC37 TI1		
65	RP 10166 FF67	}	Fill up with Z's
66	TP FC35 TI2		
67	RJ TH2 TH		Write last block
70	0 0 0		Set by FF32 to MJ 0 GG or by GG14 to MJ 0 GG16
71	TP TI4 TE37	}	Patch
72	MJ 0 FF26		
	CA FF73		Save first TI4 for forming segment table
	IA FG		Set code word for tape 3 resp. 6 with TN ≠ 0
0	SP FC40 0	}	Set code for: WRITE 1 BLOCK OF TAPE
1	AT TN TH3		
2	MJ 0 30000		
3	SP FC41 0		
4	MJ 0 FG1		Set code for: (only used in RJ) REWIND TAPE
	CA FG5		

	IA GG		Come from switch FF70
0	TP FC33 TI	}	Put TWO Δ B Δ SEGMENT Δ in TI and TI1
1	TP FC34 TI1		
2	SP TE14 0	}	Put # of entries in TI2
3	SS FC20 0		
4	AT FC5 TI2		
5	RP 10165 GG7	}	Fill rest of first block with Z's
6	TP FC35 TI3		
7	RJ TH2 TH		Write one block
10	SP TI2 0	}	Set counter for # of blocks
11	DV FC21 TE16		
12	TP A TE17		Save remainder for addr. to fill with Z's
13	TU FC4 FF37		Restore first addr. of Op File IIb in transfer command
14	TP RC3 FF70		Set switch E to E2 (MJ 0 GG16)
15	MJ 0 FF34		Jump to write rest of blocks
16	TP FC11 Q	}	Come from switch FF70. Mask 0 77777 0 → Q Have all statem. CW's been processed?
17	QT BB42 A		
20	EJ SD1 GI22		
21	MJ 0 GH		Go back to beginning for next segm. after having handled segm. table
22	TP FC35 TI	}	Write sentinel block
23	TP FC35 TI1		
24	RJ TH2 TH		

25	RJ TH2 TH	}	Write second sentinel block
26	RJ FG2 FG3		
27	RJ TH2 TH		
30	TP FC42 TH3		
31	MJ 0 BB2		Transfer parameter to read in Phase III
	CA GG32		Go to: EXIT OUT OF SEGMENTATION

	IA GH		Come from GG21. Form segment table
0	RJ GH GI15		Jump (only once) to set ST and do presettings
1	SP TE22	}	$\frac{A + 2x \# \text{ of } 77 \text{ CW's in Op File IIa}}{170}$ or $\frac{A + B}{170}$
2	AT TE37 A		
3	DV FC21 TE34		
4	ZJ GH5 GH6	}	Is there remainder? } Must be done separate because of ZJ before! Add 1 for remainder of $\frac{A + B}{170}$ Add 1 for sentinel block
5	RA TE34 FC5		
6	RA TE34 FC5		
7	SP TE22 1	}	$\frac{2x \# \text{ of } 77 \text{ CW's in Op File IIa}}{170}$ or $\frac{B}{170}$
10	DV FC21 TE26		
11	ZJ GH12 GH13		
12	RA TE26 FC5		Add 1 for remainder of $\frac{B}{170}$
13	RA TE34 TE26		Add both terms = # of blocks → TE34
14	TP TE33 Q		Mask (event. shifted) → Q
15	SP TE34 33	}	Mask the # of blocks into segm. table
16	QS A ST2		
17	RA GH16 FC5		Adv. last instr. by 1 in v
20	IJ TE35 GH32		Go to Exit (handl. next segm) Ind 178 down?
21	TP FC73 TE35		Res. ind.
22	TV GI21 GH16		Set GH16 to ST1 in v
23	LQ TE33 33		Shift mask
24	RS GH15 FC72		Reduce shift count by 11
25	IJ TE36 GH32		Go to Exit Ind 38 down?
26	TP GH32 A	}	Was this last segm.? Yes → GH32
27	EJ FC71 GH32		

30 TU EU WN

31 MJ 0 WN

32 MJ 0 BB23

CA GH33

No } alarm MORE THAN 63 SEGMENTS

Gets changed with last segm. in RJ
to MJ 0 GI23

	IA GI		Come from BB17 (only once)
0	TP 12 A		Indicator for READ, LIST, BOTH in u → A
1	EJ FC7 GI5		Zero?
2	EJ FC51 GI10		One? only LIST
3	EJ FC13 GI11		Two? only READ
4	SP FC 0		Three assumed
5	AT FC67 ST		Add space for Tape Hd1 + Contr + 1 + Term buffer → ST
6	TV A 12		Set v part of 12
7	MJ 0 GL		After ST set, do preliminary settings
10	RS GI11 FC51		Case only LIST
11	SP FC2 0	}	Case only READ
12	MJ 0 GI5		
13	TP TE40 TE22		Come from EI set # of 77 CW's for forming segm.
14	MJ 0 ET		Go to setting BB42 (RJ exit of ET3 is already set to EI1)
15	TP FC70 TE33		Come from GH (only once) put mask in TE33
16	TP FC73 TE35		Set ind. for first time 17 (20 rows)
17	TP FC25 TE36		Set ind. 3 (4 words per row)
20	RP 10020 GH1	}	Clear rest of segm. table and jump to GH1
21	TP FC7 ST1		
22	RJ GH32 GH	}	Come from GG20
23	MJ 0 GG22		
	CA GI24		Case we have last segment

	IA GJ		Come from FF
0	TP ST A	}	Set once for whole program N in TE41 to FN + 1 for FN + 1 + T ₂ (later formed)
1	AT FC5 TE41		
2	MJ 0 30000		Used only once in RJ from FF
	CA GJ3		
	IA GL		Come from GI7
0	SP 7 25	}	Set TE21 segment length
1	LT 0 A		
2	ST ST TE21		
3	RS TE21 FC6		
4	MJ 0 BB11		
	CA GL5		

	IA HH		Come from CC11	Case CW 4---	
0	TU TE2 HH3	}			
1	RA HH3 FC10				
2	SP TE20 0			Is this 4--- CW already in Op File IIa?	
3	RP 30000 HH7			No	
4	EJ FA2 HH5	}		Yes	
5	RA TE11 FC5			Adv. to next cross ref.	
6	MJ 0 BB64			See whether all handled?	
7	TU BB73 HH10	}			
10	RP 30000 HH12			Search	
11	EJ FD1 HH14	}		Found	
12	TU FU WN			Not found	
13	MJ 0 WN			Go to alarm	
14	SN Q 17	}		} search 4--- in Directory I and store drum address (Op File I) in TE26(v)	
15	SA HH10 0				Found, put drum address (where CW placed in Op File IIa) → TE26
16	SA HH11 0				
17	TU A HH20				
20	TP 30000 TE26	}			
21	TV TE5 HH22			Place CW in Op File IIa	
22	TP TE20 30000	}			
23	RJ CE1 CE			Adv. by 2 in v and check exceeded region	
24	RA TE2 FC13			Adv. by 2 in u	

25	TP FC12 Q	v mask → Q	} Adv. to next wanted Op File I address and put addr. in v of TE26 Put CW in u of TE20
26	LA TE26 A17	Addr → u of A	
27	TU A HH30	v of word in Directory I → A	
30	QT 30000 A		
31	AT TE26 TE26	Addr. for next CW → TE26	} Next CW → u of TE20
32	LA A 17		
33	TU A HH34		
34	TU 30000 TE20		
35	TP FC3 Q	Mask 0 77000 0 → Q	} Is next CW 23----? 4----? Yes, exit this part
36	QT TE20 A	Is CW 23----?	
37	EJ FC56 HH5		
40	TP FC54 Q	Mask 0 70000 0 → Q	
41	QT TE20 A	Is CW4 - ?	} Is CW already in Op File IIa ?
42	EJ FC55 HH5		
43	TU TE2 HH46	None of them	
44	RA HH46 FC10		
45	SP TE20 0		} Is CW already in Op File IIa ?
46	RP 30000 HH21		
47	EJ FA2 HH25		
	CA HH50		

IA RC

Constants for switches

0 MJ 0 FF1
 1 MJ 0 FF12
 2 MJ 0 GG
 3 MJ 0 GG16
 4 RP 30000 FF52
 5 RP 10000 FF60

CA RC6

IA WN

Alarm routine

0 TP 30000 UP3
 1 RJ UP2 UP
 2 MJ 0 BB1

CA WN3

IA BU

0 0 BU1 0
 1 0 BU2 12
 2 51 50300 16566
 3 24 66304 73050
 4 66 01713 46633
 5 01 24464 60154
 6 30 31305 43050
 7 26 30650 13465
 10 01 66515 10146
 11 24 54323 00131
 12 51 54016 53032
 13 47 30506 62277

Alarm: ONE STATEMENT WITH ALL
 REFERENCES IS TOO LARGE FOR SEGMENT.

O N E Δ S T
 A T E M E N
 T Δ W I T H
 Δ A L L Δ R
 E F E R E N
 C E S Δ I S
 Δ T O O Δ L
 A R G E Δ F
 O R Δ S E G
 M E N T .

CA BU14

	IA	EU	
0	0	EU1	0
1	0	EU2	4
2	47	51543	00166
3	33	24500	11106
4	01	65303	24730
5	50	66652	27777

Alarm: MORE THAN 63 SEGMENTS.

	CA	EU6	
	IA	FU	
0	0	FU1	0
1	0	FU2	11
2	52	65306	72751
3	01	51523	05424
4	66	34515	00134
5	65	01543	03130
6	54	30502	63027
7	01	25676	60127
10	51	30650	15051
11	66	01245	25230
12	24	54227	77777

Alarm: PSEUDO OPERATION IS REFERENCED BUT DOES NOT APPEAR.

	CA	FU13	
--	----	------	--

	IA FC	Constants
0	0 0 344	$5 + 103_{10} + 120_{10}$ for both
1	0 0 175	$5 + 120_{10}$ for list alone
2	0 0 151	$2 + 103_{10}$ for read alone
3	0 77000 0	
4	0 FP 0	
5	0 0 1	
6	0 0 2	
7	0 0 0	
10	0 20000 0	
11	0 77777 0	
12	0 0 77777	
13	0 2 0	
14	0 0 DI	
15	0 0 DI1	
16	0 FD1 0	
17	0 0 FA2	
20	0 0 FP1	
21	0 0 170	
22	0 170 0	
23	RJ DA11 DA	} Used for patch of Phase I, ML1
24	MJ 0 TT	
25	0 0 3	
26	0 FA2 0	
27	31 34463 00101	
30	66 71510 10101	

31 66 71510 12401
32 65 30324 76601
33 66 71510 12501
34 65 30324 76601
35 74 74747 47474
36 30 50270 15131
37 01 30506 65473
40 71 00103 TI
41 10 3 0
42 50 1 1100
43 0 0 22
44 0 0 24
45 0 0 30000
46 0 26000 0
47 0 0 DI2
50 0 0 DD12
51 0 1 0
52 0 0 BB46
53 0 27000 0
54 0 70000 0
55 0 40000 0
56 0 23000 0
57 0 0 4
60 0 0 TE31
61 MJ 0 EF
62 0 0 EE1

63 0 0 S0
64 0 0 SM
65 0 0 SN
66 0 0 5
67 0 0 GK
70 77 70000 00000
71 MJ 0 GI23
72 0 0 11
73 0 0 17
74 0 TE3 ER
CA FC75

Temporaries TE

		Segment Phase II	
TE0		T	# of addresses (generated rout.) in segm. for 22, 24, 25, 26, 27, 40, 50, 77
1		t	# of addresses per sentence; set with TV in BB31
2	K		# of entries in Op File IIa
3		T ₂	# of addr. for 26, 27 only; adv. in EE24 T ₂ + t ₂ → T ₂
4		t ₂	# of addr. per sentence; set with TV together with TE1
5		α	Present statem. CW addr. of Op File IIa
6		β	Next statem. CW addr. of Op File IIa
7		γ	Next statem. CW addr. of Op File IIb
10	[]	TEMP 1	Statem. CW(u) + # of words in item (v) set with TP
11		TEMP 2	Addr. of cross ref. CW; set by ...
12		TEMP 3	# of 77--- data words in this segm.; adv. by 1 with RA
13		TEMP 4	Op File IIa addr. for this statem. CW
14		TEMP 5	Op File IIb addr. for last cross ref. of previous statem. CW
15		INDEX 1	Index for # of cross ref; set with TV
16		INDEX 2	# of full blocks to be written and used by EN, EQ for storage
17		R	# words in partial block
20	[]	WS	Holds CW to be placed, whose addr. is in TE11
21	[]	[]	Segm. length for problem; fixed for whole problem

22	[]	Counter for # of 77 CW's (CW's, not addresses)
23	[]	Op File IIa address of first statem. CW in Vary loop
24		t ₄	
25		t ₅	
26	[]	Storage space used by EF3, BB10, HH, GH
27		t ₁	# of instructions for 1 sentence at a time
30		t ₃	Index that we are in Vary within Vary
31	[]	CW of last statem. in Vary loop for comparison
32	[]	Storage for γ in EG and EH part
33	[]	77 70000 00000 mask shifted for segm. table
34	[]	Quotient for GI and GH for segm. table
35	[]	Index for 17g for segm. table
36	[]	Index for 3 for segm. table
37	[]	Storage for TI4 of first block for segm. table
40	[]	Storage for reset TE22
41	[]	Storage for N + 1

2. ALLOCATOR

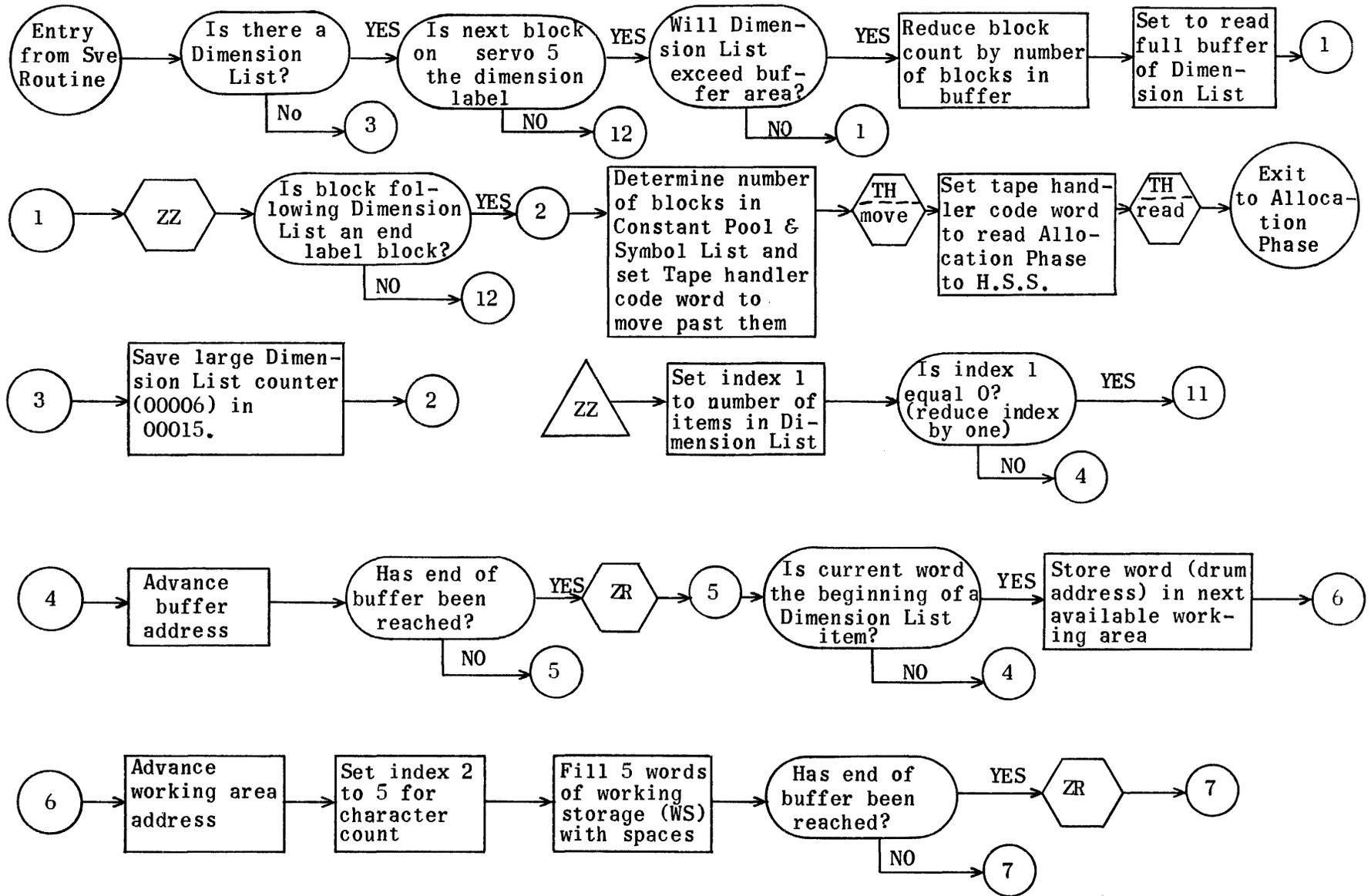
2. Allocator

a. ALLOCATION Setup

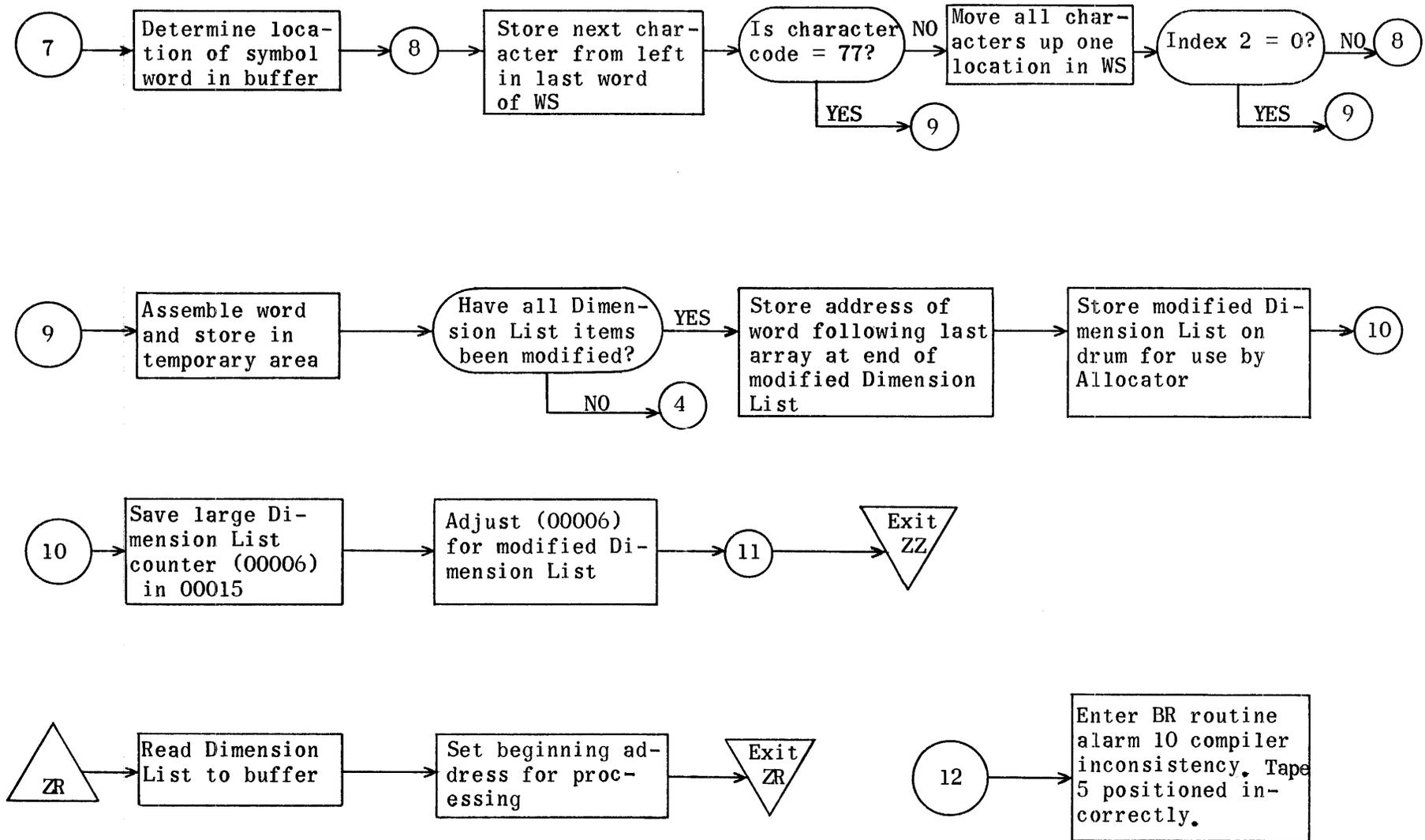
The setup routine for the Allocator reads the Dimension List from magnetic tape and modifies it so that each array is represented by two words instead of the variable (up to six) word items of the original list. The modified Dimension List is then stored on the drum for use by the Allocator. The Dimension List is modified at this time because the Allocator and later the Processor make extensive use of drum storage. Between these phases, the Initialization Generator must have more Dimension information than is available in the modified Dimension List so the original Dimension List is read again from tape and stored on drum preceding the operation of that phase.

After modifying the Dimension List, the setup routine adjusts the Dimension List counter (at location 00006) to reflect the length of the modified list. The counter for the original list is saved at location 00015. The tape on Uniservo 5 is then moved forward past the Constant Pool and Symbol List so that it is positioned properly for the Allocator to write Op File III, Preface, and Termination.

The seven blocks of the Allocator are then read from the UNICODE Master Tape and control is transferred into the Phase.



Allocation Set-Up Flow Chart



Allocation Set-Up (cont.)

Region for Allocation Setup

RE	DC22	Buffer load (in blocks)
RE	TH21	Tape handler
RE	DD40101	Modified Dimension List
RE	BR537	Compiler Inconsistency Routine
RE	ZI7230	
RE	ZZ7270	
RE	ZR7354	
RE	ZX7362	
RE	ZT7403	(1) Temporary (holds number of blocks of Dimension List)
RE	WS674	Working area
RE	ZY2705	Buffer area (= WS 2011)
RE	LC674	Storage and execution address of Allocator
RE	MA700	7 = # blocks Allocation phase?
RE	LA7064	LA = last word of buffer area.

Allocation Setup

	IA	ZI		
	0	TP	ZX	Q
	1	QT	14	A
	2	ZJ	ZI3	ZI24
	3	LT	11	A
	4	ST	ZX1	ZT
	5	TP	ZX2	TH3
	6	RJ	TH2	TH
	7	TP	ZY	A
	10	EJ	ZX3	ZI12
	11	MJ	0	BR12
	12	SP	ZT	0
	13	TJ	ZX4	ZI16
	14	RS	ZT	ZX4
	15	TU	ZI13	ZR
①	16	RJ	ZZ63	ZZ
	17	TP	ZX2	TH3
	20	RJ	TH2	TH
	21	TP	ZY	A
	22	EJ	ZX17	ZI25
	23	MJ	0	BR12
③	24	TP	6	15
②	25	SP	14	0
	26	LT	3	A

Is there a Dimension List?

blocks Dimension List → temp.

Read one block and check label for DIMENS.

Tape #5 positioned incorrectly.

Will Dimension List exceed buffer area?

Reduce block count by buffer length.

Set to read full buffer of Dimension List.

To build modified Dimension List.

Read one block and check label for E N D Δ O F

Tape #5 positioned incorrectly

Large Dimension List counter → 15.

blocks in Constant Pool

27	LQ	14	Q17	}	Add # blocks in Symbol List
30	QA	ZX13	A		
31	SP	A	25	}	Move Uniservo #5 forward past Constant Pool and Symbol List.
32	AT	ZX15	TH3		
33	RJ	TH2	TH	}	Read Allocation Phase to H.S.S.
34	TP	ZX16	TH3		
35	RJ	TH2	TH	}	Jump to Allocation Phase.
36	MJ	0	LC		
	CA	ZI37			

	IA	ZZ			
	0	TV	6	ZZ3	Set index to # items in Dimension List
	1	IJ	ZZ3	ZZ6	Reduce index by one
	2	MJ	0	ZZ63	To exit
	3	0	0	0	Index
④	4	RA	ZZ10	ZX11	Advance address within buffer.
	5	TJ	ZX12	ZZ7	Still in buffer?
	6	RJ	ZR5	ZR	No; so go to read in next block
⑤	7	TP	ZX6	Q	Bit 29 mask → Q
	10	QT	30000	A	Bit 29 of word → A
	11	ZJ	ZZ12	ZZ4	If = 0, recycle
	12	TU	ZZ10	ZZ13	} If not = 0, store first word (drum address)
	13	TP	(30000)	(WS10)	
⑥	14	RA	ZZ13	ZX7	} Advance to next word
	15	TV	ZZ13	ZZ40	
	16	RA	ZZ13	ZX7	Set address of 1st word for next variable
	17	TP	ZX10	WS	Set index for character count.
	20	RP	10005	ZZ22	} Fill with XS3 spaces
	21	TP	ZX7	WS2	
	22	RA	ZZ10	ZX11	} Text for end of buffer area
	23	TJ	ZX12	ZZ25	
	24	RJ	ZR5	ZR	Read in 2nd buffer load
⑦	25	TU	ZZ10	ZZ26	Set to address of symbol word
⑧	26	LQ	30000	6	
	27	QT	ZX13	WS7	One digit → WS7

9

30	EJ	ZX13	ZZ34	If = 77, go to assemble digits
31	RP	30006	ZZ33	If ≠ 77, move digits up one word in the image.
32	TP	WS2	WS1	
33	IJ	WS	ZZ26	6 digits?
34	SP	WS1	6	} Assemble digits in A _R
35	RP	20004	ZZ37	
36	SA	WS2	6	
37	SA	WS6	0	
40	TP	A	30000	Store assembled word.
41	IJ	ZZ3	ZZ4	Recycle to get all arrays
42	TV	ZZ13	ZZ53	} Set addresses to calculate and store final word
43	RS	ZZ40	ZX7	
44	TV	A	ZZ52	
45	TP	ZZ2	ZZ25	
46	RJ	ZZ25	ZZ22	
47	RJ	ZZ25	ZZ22	
50	TU	ZZ10	ZZ51	
51	TU	30000	ZZ3	
52	RA	ZZ3	30000	} Form and store final word.
53	TP	A	30000	
54	RP	32001	ZZ56	} Store modified Dimension List.
55	TP	WS10	DD	
56	TP	6	15	Save counter for large dimension list.
57	TP	ZX14	Q	} Adjust (00006) for modified Dimension List.
60	SP	6	20	
61	SA	ZX11	0	

11

62	QS	A	6
63	MJ	0	30000
CA	ZZ64		
	IA	ZR	
0	SP	[ZT]	25
1	AT	ZX5	TH3
2	RJ	TH2	TH
3	TU	ZX20	ZZ10
4	TU	ZI12	ZR
5	MJ	0	30000
	CA	ZR6	

Read Dimension List to buffer

Set beginning address for processing

	IA	ZX		
0	07	70000	0	
1	0	0	2	
2	50	105	ZY	
3	27	34473	05065	D I M E N S
4	0	0	DC	
5	50	5	ZY	Read Parameter (except # blocks.)
6	0	40000	0	
7	0	0	1	
10	0	0	5	
11	0	1	0	
12	QT	LA1	A	LA = last word of buffer area
13	0	0	77	
14	0	7777	0	
15	30	5	0	
16	50	MA1	LC	
17	30	50270	15131	E N D Δ O F
20	0	ZY	0	
	CA	ZX21		

b. ALLOCATION PHASE

The Allocation Phase serves two purposes:

- 1) Builds Op File III for each segment and writes on tape.

Op File III (2 word items)

	<u>Op</u>	<u>u</u>	<u>v</u>	
1.		Call word		
2.		# lines in routine	H.S.S. running lo-	
		or		
1.		Call word		
2.	14	Segment # from	Segment # jumped to	H.S.S. running lo-
		or		
1.		Call word		
2.		# of data	H.S.S. running lo-	

If the end point of one-way jump is in another segment

If call word is of the form 77xxx which refers to a group of data, e.g., x₁, x₂, ... x₁₀.

- 2) Generates the necessary instructions to manipulate data between segments during the running program. These instructions are called:

- a) The Preface, which transfers 77xxx type data to their storage locations in H.S.S.
- b) The Termination, which transfers updated 77xxx type data to their designated locations on MD.

The Preface and Termination instructions operate in H.S.S. during the interlude between 2 segments. After generation of these instructions for each segment, the Preface and Termination are written on magnetic tape.

Input: The Allocator receives as input (from Segmentation):

- 1) Op File IIa - call words of routines and data in segment.
- 2) Op File IIb - call words of end points of all one way jumps within the segment.

These files are on Uniservo tape by segment.

- 3) Dimension List with MD storage addresses for 77xxx data.

Output: The output of Allocation consists of:

- 1) Op File III by segments on tape.
- 2) Preface and Termination for each segment on drum.

Procedure: Read Op Files IIa and IIb into H.S.S. one segment at a time. Then compare each call word in Op File IIb against the entire Op File IIa for this segment to determine if the end of the jumps (which are actually the words in IIb) appear in the same segment. If equality is not met, the call word from IIb is entered in IIa, thus increasing the length of Op File IIa. Each new entry into IIa at this time is accompanied with the flag 14 in the operation position of the next word. Thus, each new entry in IIa is an entry of 2 words. Each time an entry is made in Op File IIa the call word from IIb is also placed in another list, called Directory 4, which will be used only during this phase. Each entry in Directory 4 is also a 2 word entry, consisting of call word in the first word and the segment number in the second word. An item in Directory 4 at this time looks like this:

	Op	u		v
1st word	00	Call word		00000
2nd word	00	0	Segment number	00 00000

The above procedure is followed until all the call words in Op File IIb have been checked against Op File IIa for one segment.

Each call word in Op File IIa is then checked to determine the type of routine or data to which it refers.

The determination of the type of routines used in the segment, along with the number of lines in the running routine (available in Op File IIa), enables us at this time to assign actual operating addresses according to the High Speed Storage layout:

	CONTROL SECTION (fixed length all problems; includes Tape Handler)
	BUFFER AREAS for Input-Output Instructions (as required)
S	STATEMENTS
R	SUBROUTINES 1) Library Routines 2) Pseudo Operations 3) Defining Equations
D	DATA AREA 1 Multiple valued (77---type)
	DATA AREA 2 Single-valued variables (fixed length for all segments)
	CONSTANT POOL (fixed length for all segments)

Control being of fixed length and buffer area requirements for this problem being known, we can locate S exactly. During Segmentation, a separate tally of statement lengths permits determination of R exactly. D is determined by the accumulated tally of total statement and subroutine lengths plus two. (The plus two accounts for the locations required by the Processor to provide continuity between sequential segments.) With these starting points

S, R, and D, assignment of memory locations in a forward direction can be made according to the category determined by the call word.

The number of lines of data, or the number of lines in the routine, is also used to fill in the u portion of the items in Op File IIa. At this time, Op File IIa is beginning to resemble the new Op File III which is actually an expanded and modified Op File IIa.

After completion of the foregoing process for each segment, that segment's Op File III (Formerly Op File IIa) is written on the drum, and Directory 3 is constructed, containing one word for each segment, in the following format:

Op	u	v
00	MD location of 1st word of Op File III	# of words in Op File III for this segment

Thus, the first word in Directory 3 refers to the first segment, the second word, the second segment, etc.

When Op File III for the last segment has been written on the drum, Op File III is in its final form for all items except those referring to jumps to other segments. But Directory 4 is actually a combined listing of these call words for all segments. So, we use the items of Directory 4 to search against Op File III (by segment) and fill in Directory 4 with number of the segment in which the call word is found, and the operating address of the routine during execution. This continues until all the entries in Directory 4 have been processed. A complete Directory 4 item is of the form:

	Op	u	v
1st word	00	Call word	00000
2nd word	14	0	Segment from Segment to H.S.S. running address in segment to

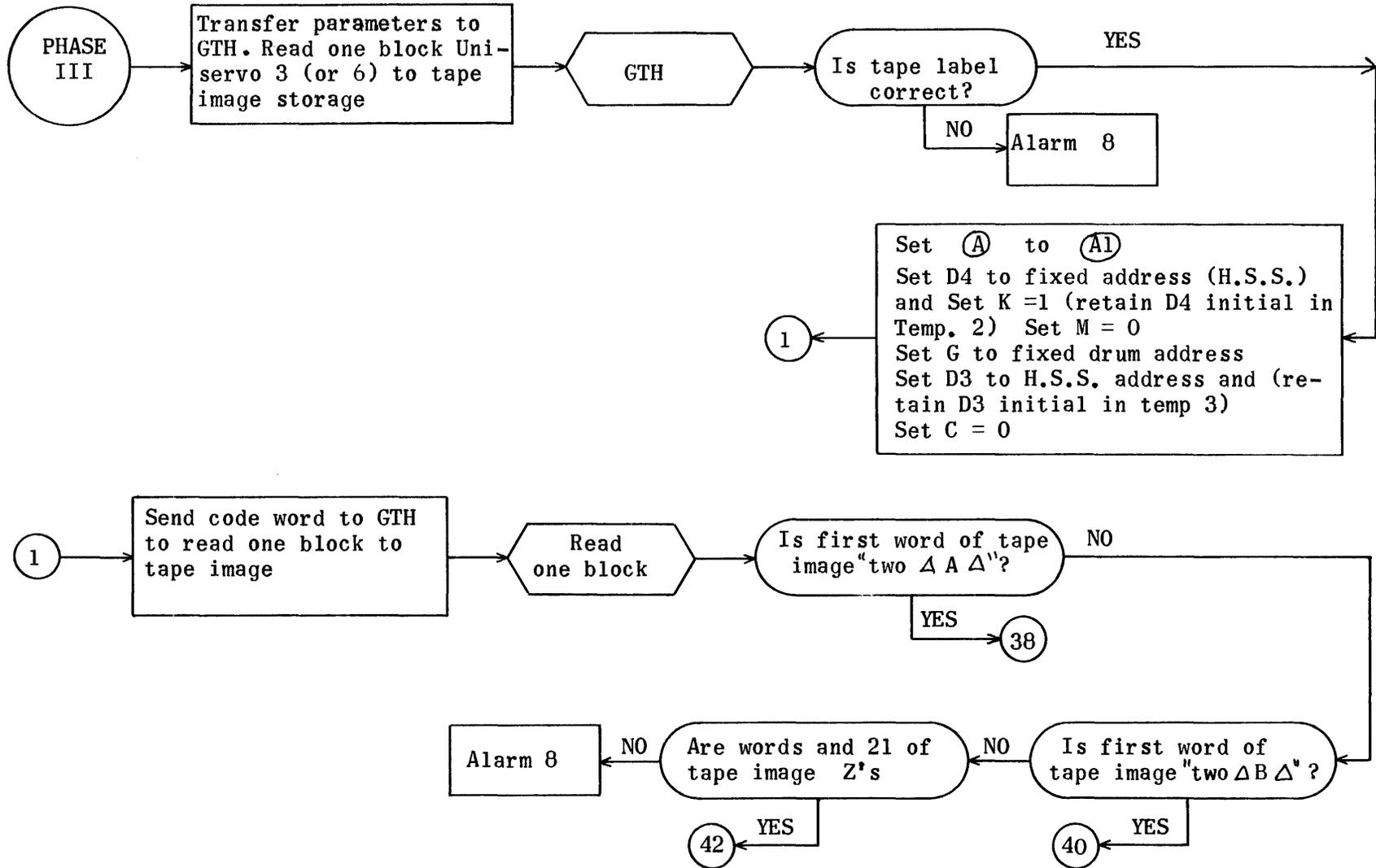
The second word of the above item in Directory 4 is filled into Op File III (one segment at a time) in its appropriate place to complete Op File III. While each segment is in H.S.S. at this time, the instructions for data manipulation are generated and stored on the drum.

The instructions for data manipulation are prepared from Op File III. Each multiple word data group has been assigned an area on MD and the starting address of the area for each variable is available in the Dimension List. Using Op File III and Dimension List information for each 77xxx type call word, the Repeated TP's are generated. When this listing is complete, the w's of Repeat orders are determined and recorded. (Reverse direction for Preface; forward for Termination.) The w's for the Preface are fixed H.S.S. operating locations (not relative) since they are generated at a point during compilation when the exact starting address of Data Area 1 (77 - - - type) is known. Since the length of the Preface is known when Termination w's are written, they too are assigned fixed addresses in the 120-word buffer area within the Control Section.

The completed Op File III and the Preface and Termination for each segment are stored on magnetic tape and will be used during the Processing Phase.

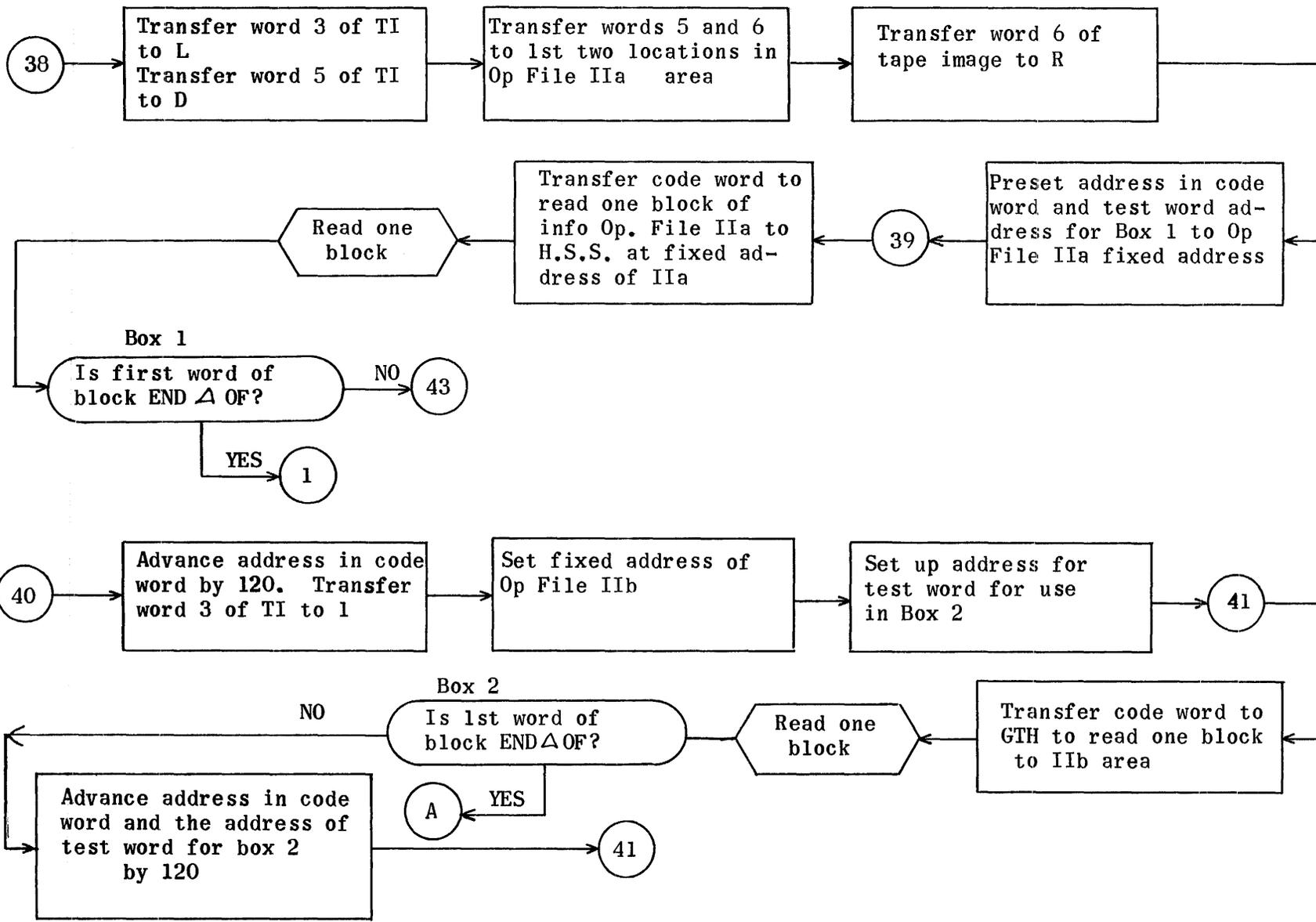
This phase is complete when Op File III, Preface, and Termination for all segments of the problem have been written on tape.

Allocation Phase Flow Chart



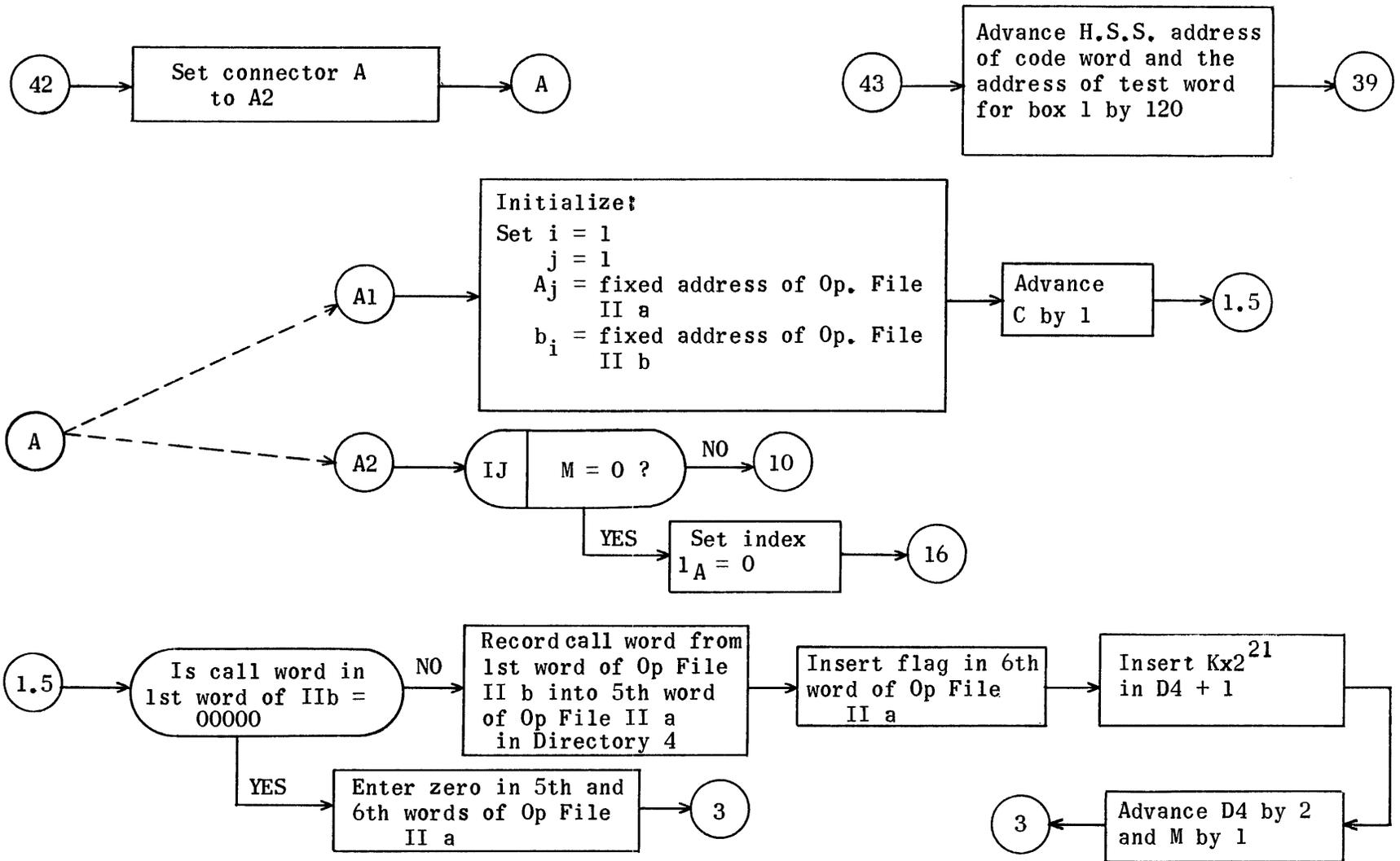
1566

Allocation Phase Flow Chart (cont.)



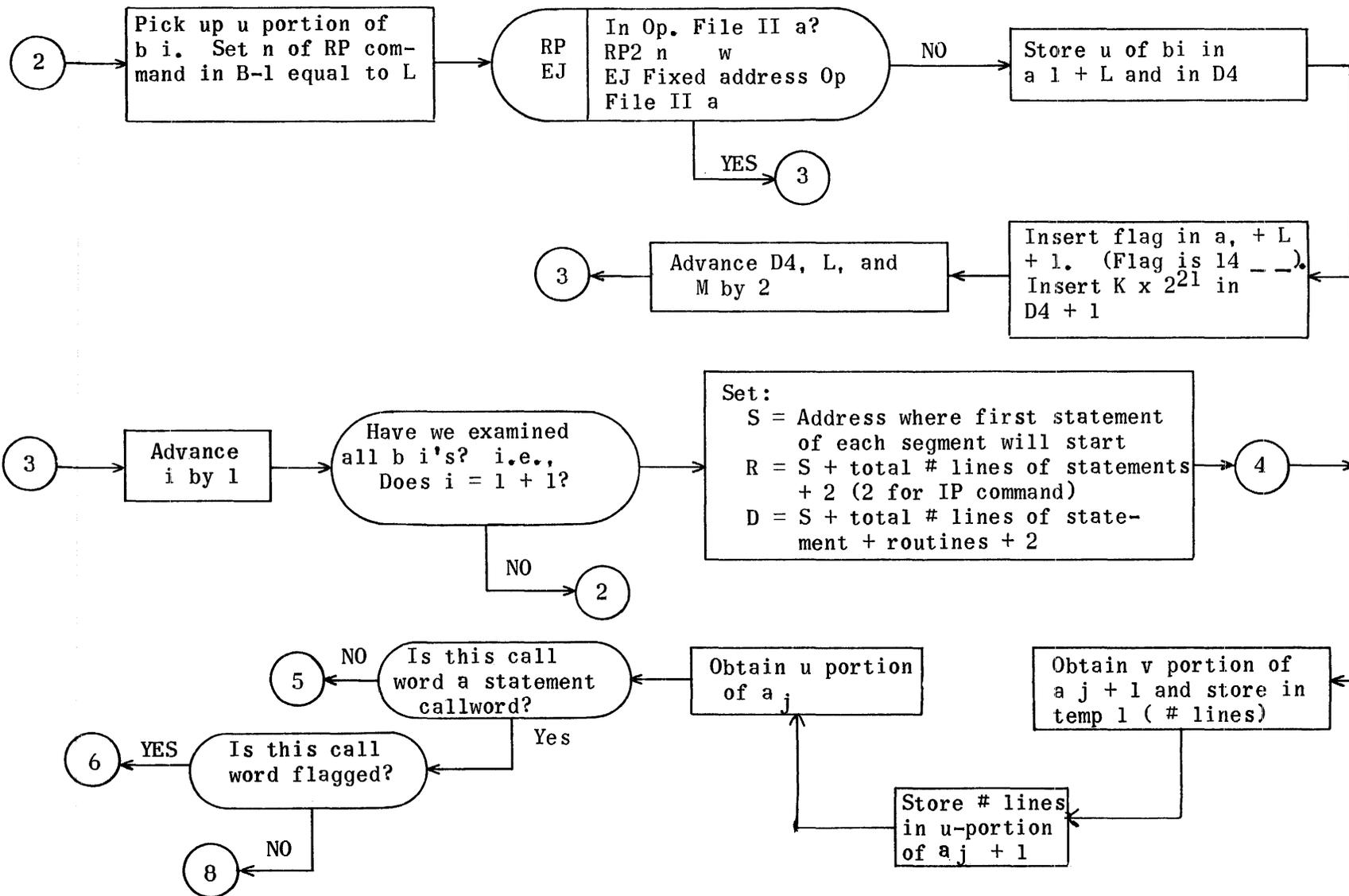
1567

Allocation Phase Flow Chart (cont.)

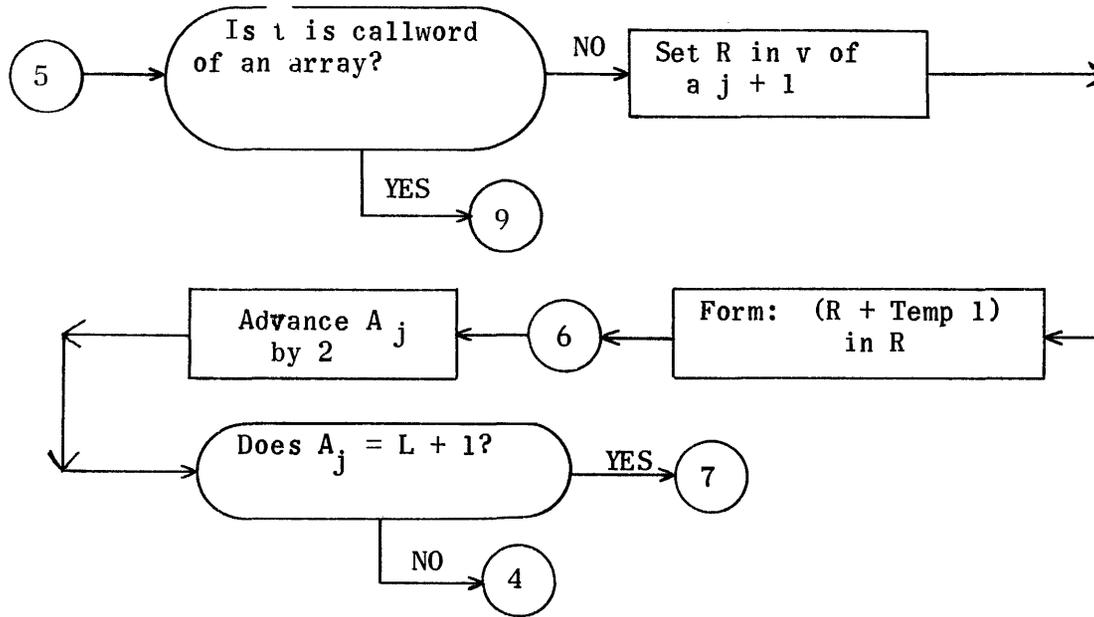


Allocation Phase Flow Chart (Cont.)

B-1



Allocation Flow Chart (cont.)



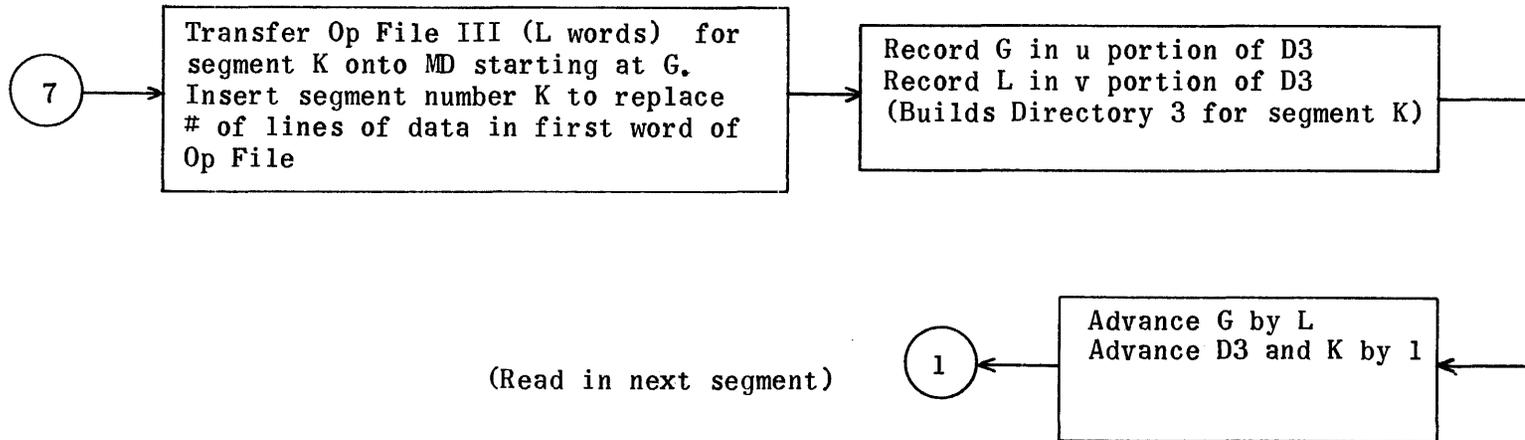
NOTE

Call words changed from:

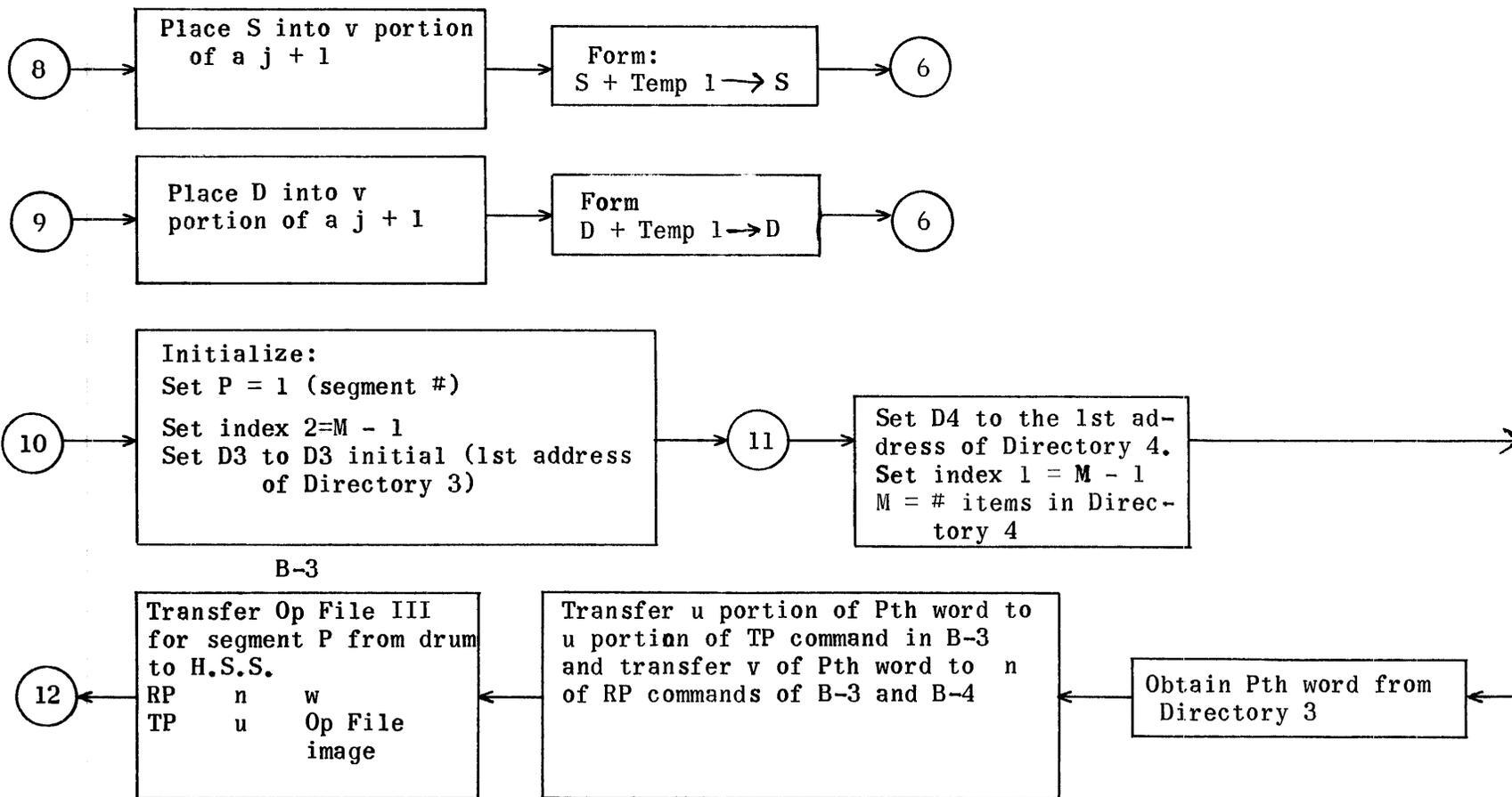
70	---	to	27	---
71	---		{	24
				25
72	---			4
73	---			5
74	---			22
75	---			26
76	---		{	66
				65
				64
77	---			77

1570

B-2

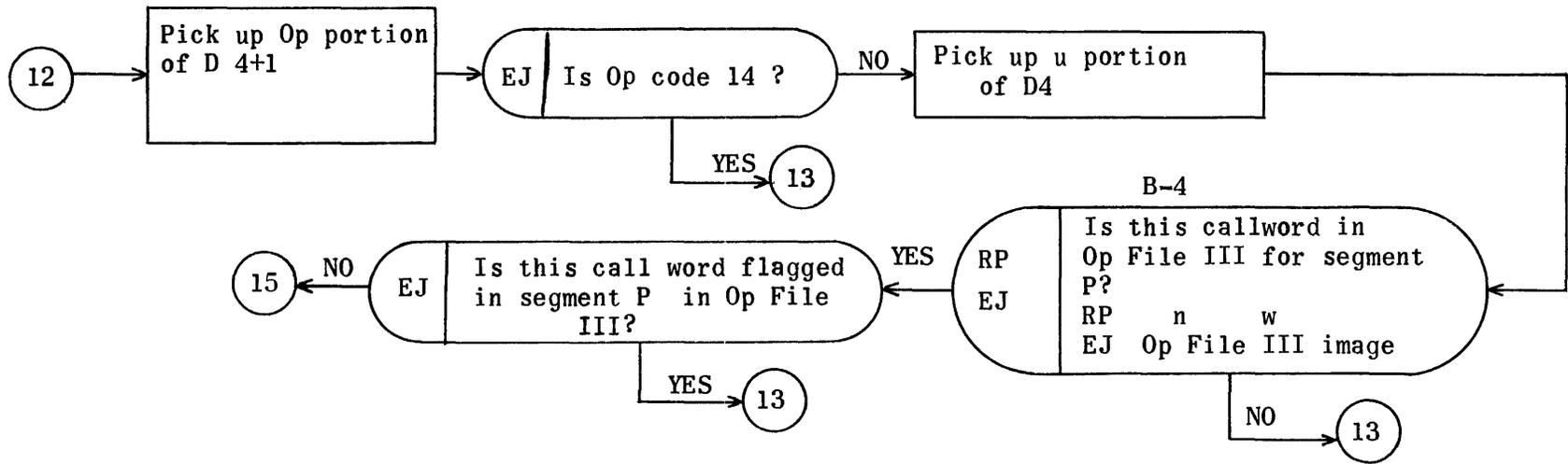


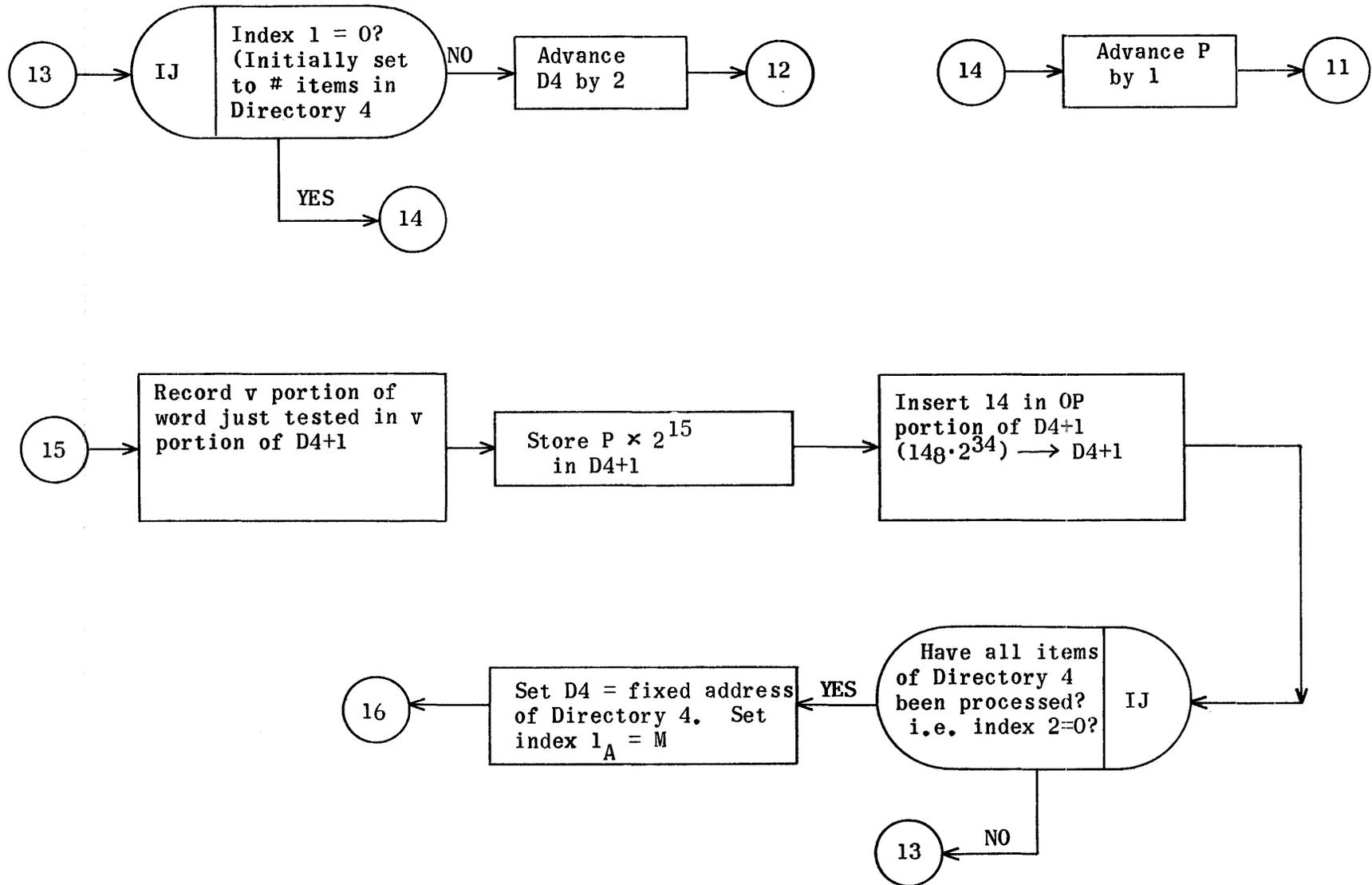
Allocation Flow Chart (Cont.)

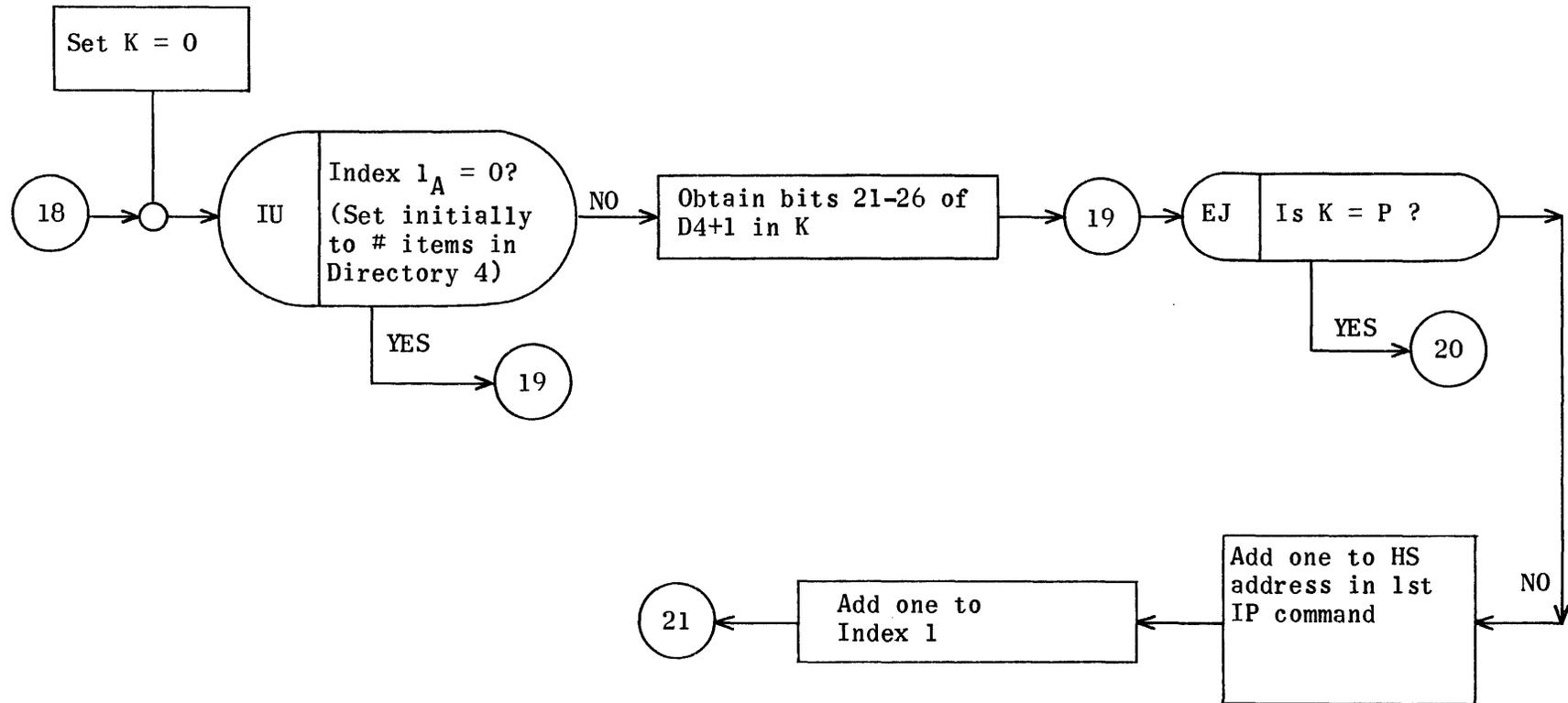
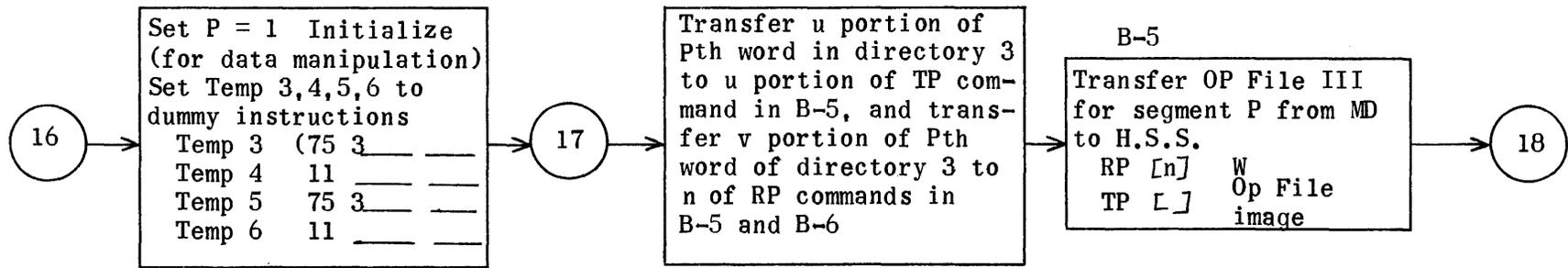


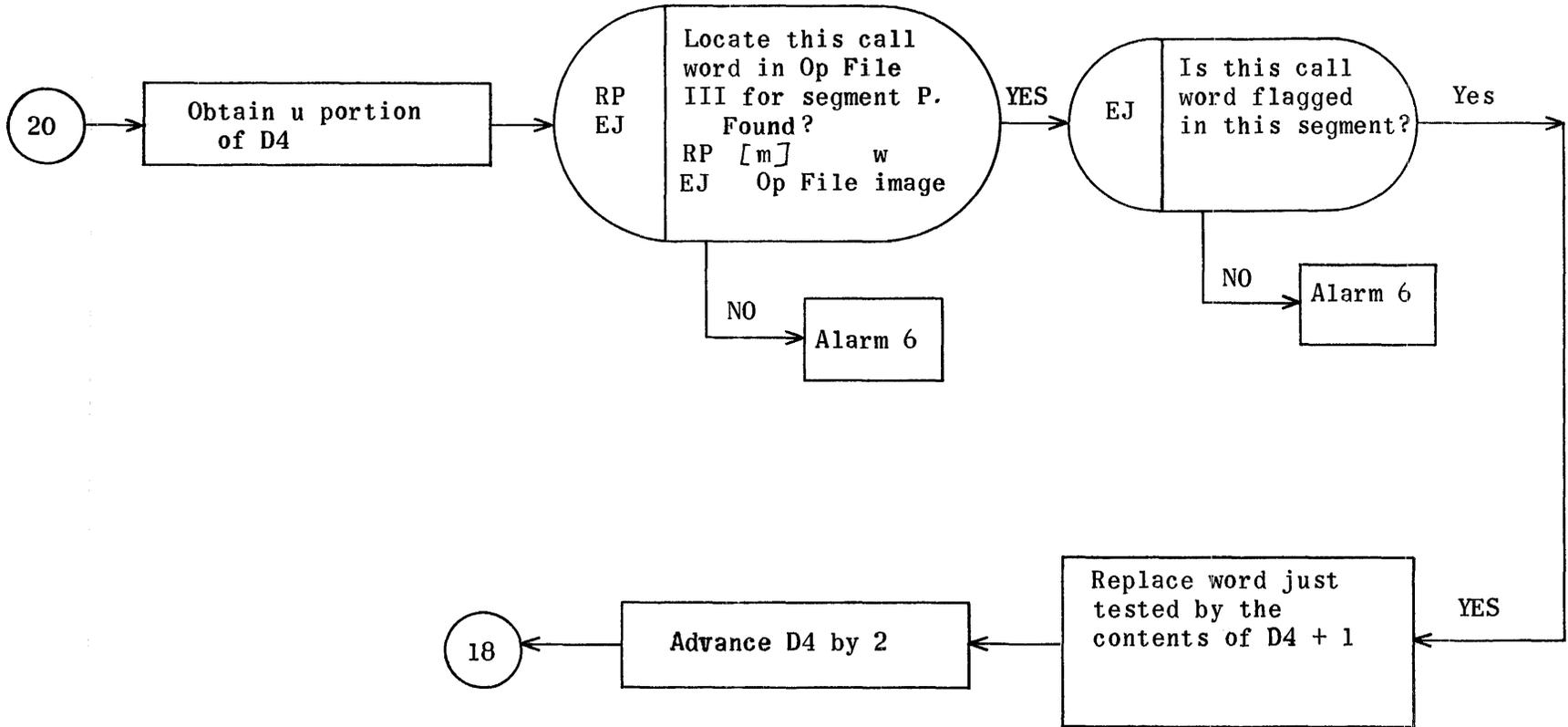
B-3

Allocation Flow Chart (Cont.)

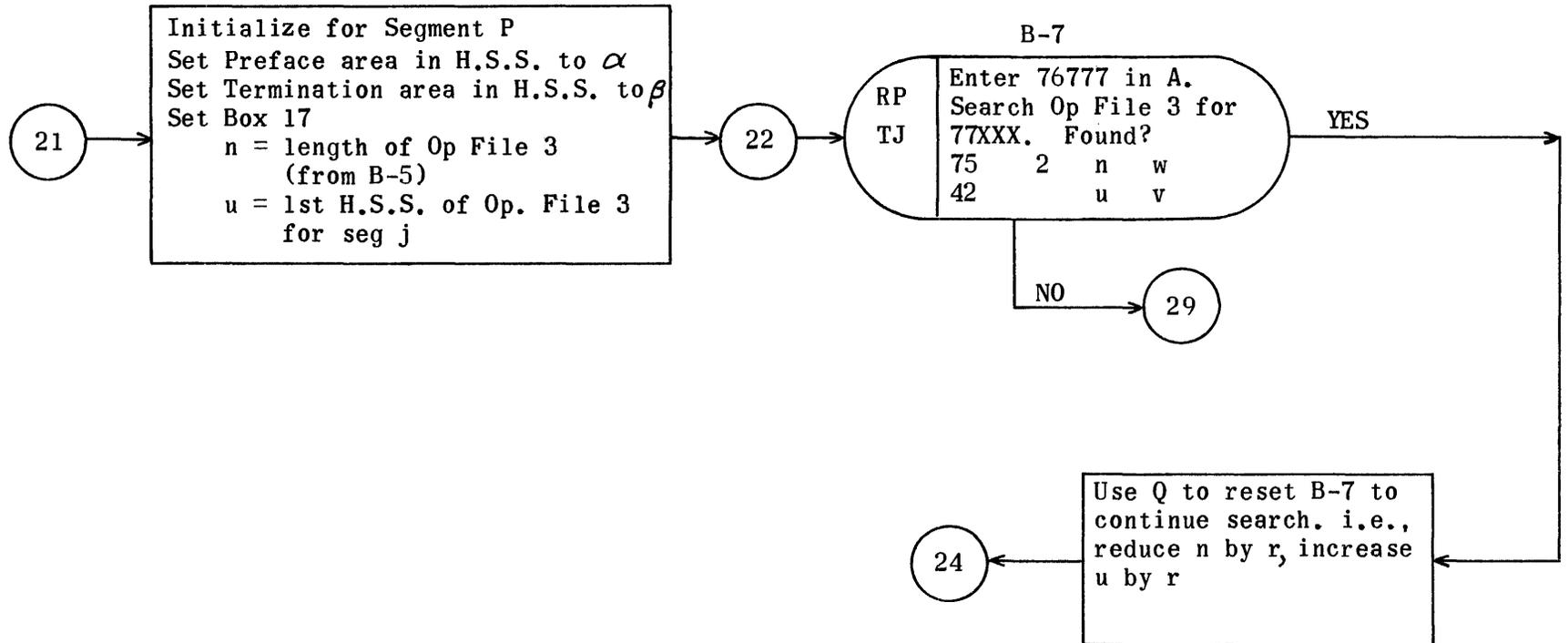




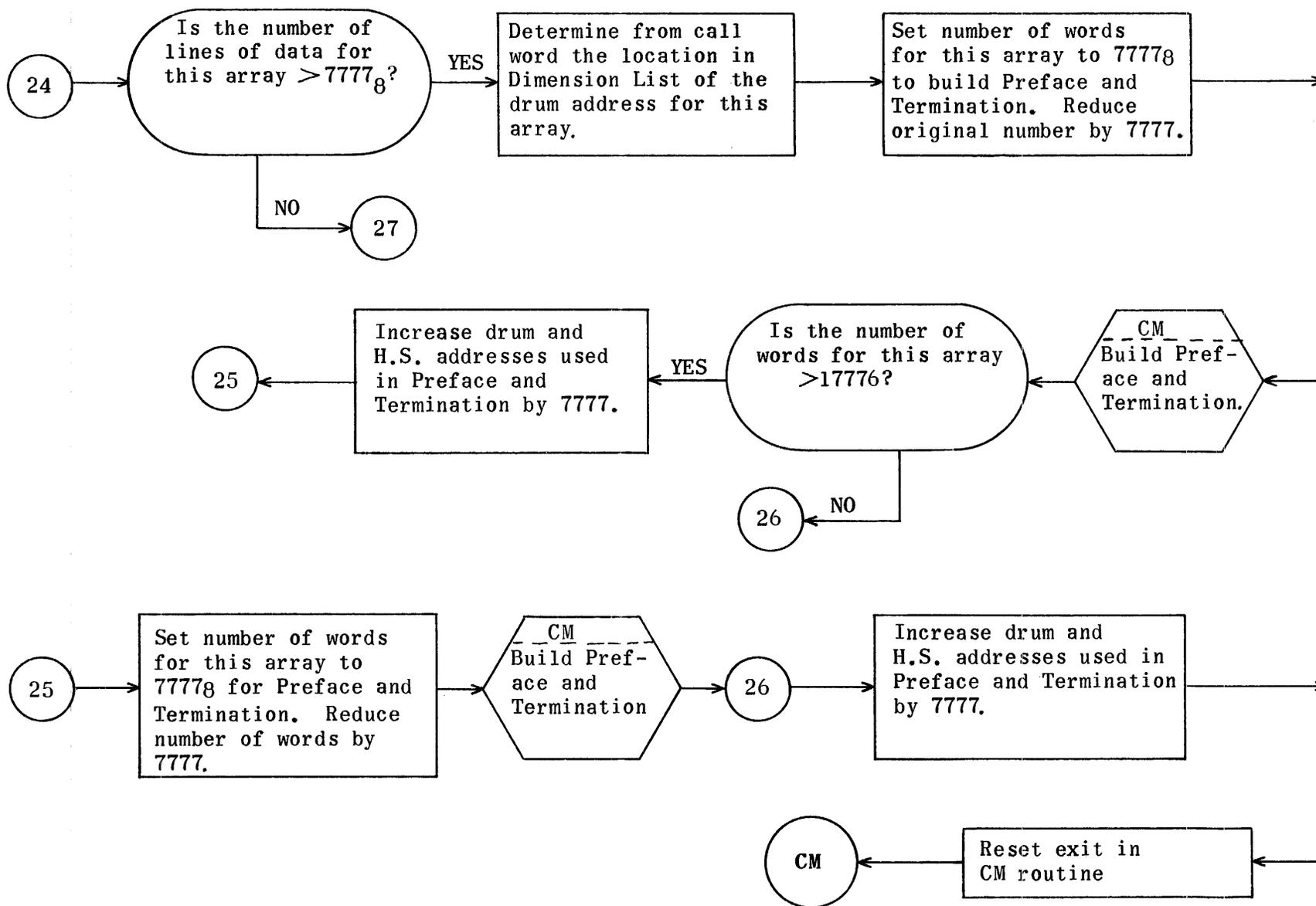


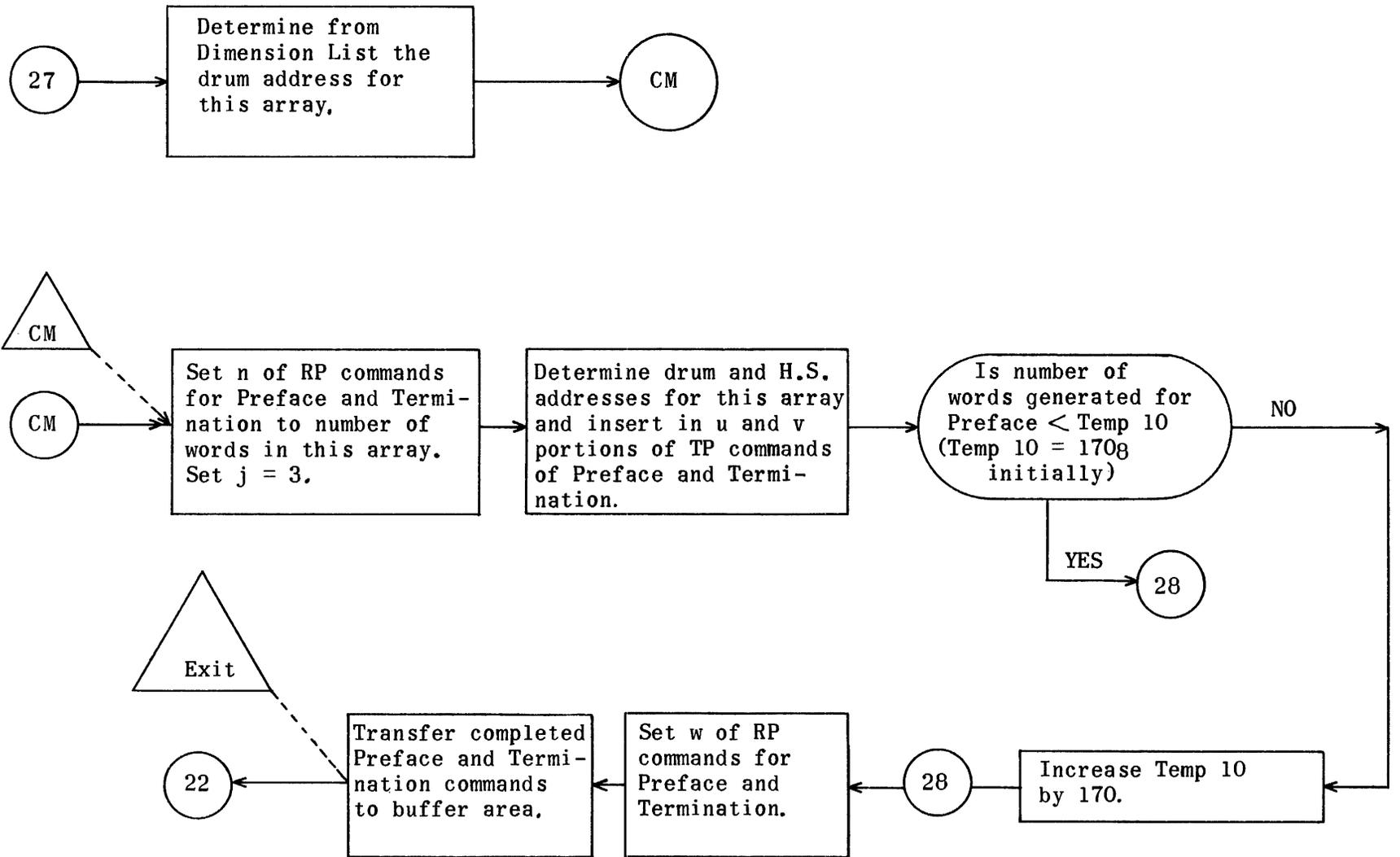


Allocation Flow Chart (cont.)

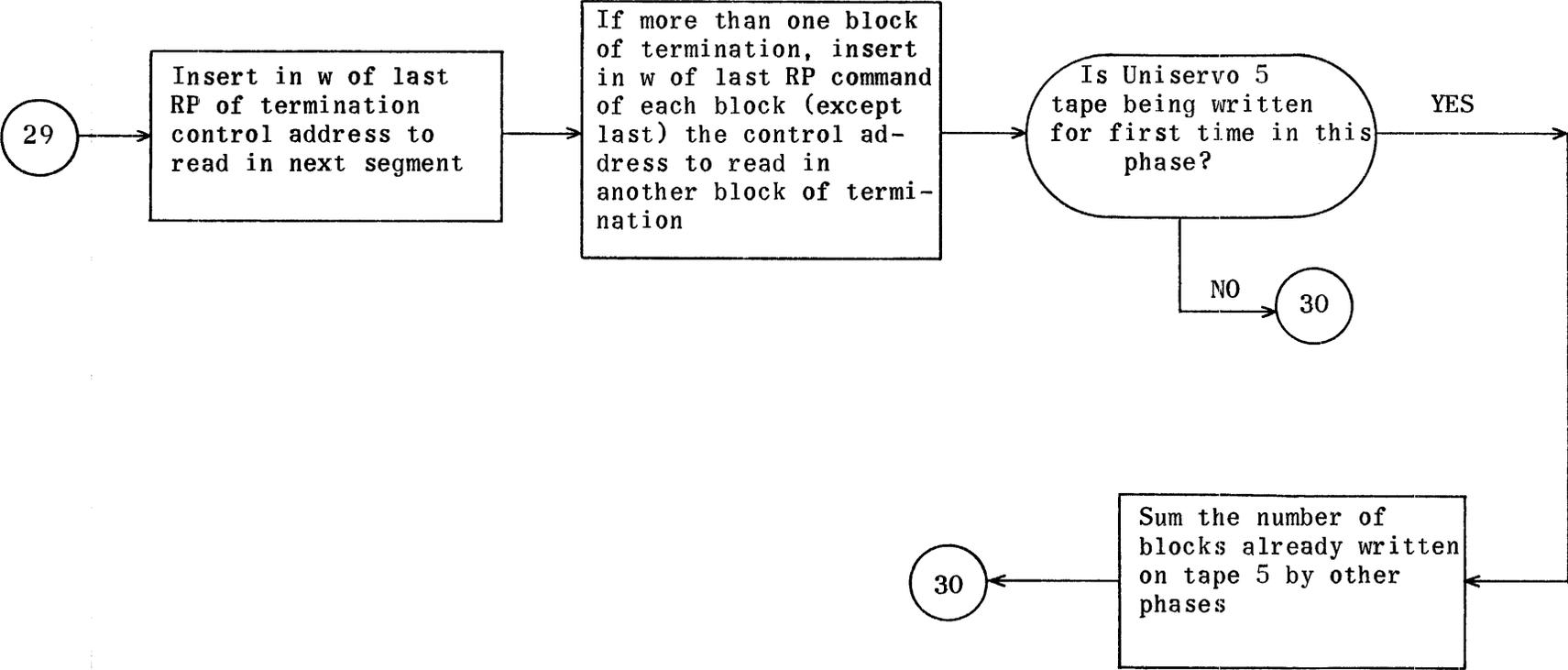


Allocation Flow Chart (Cont.)

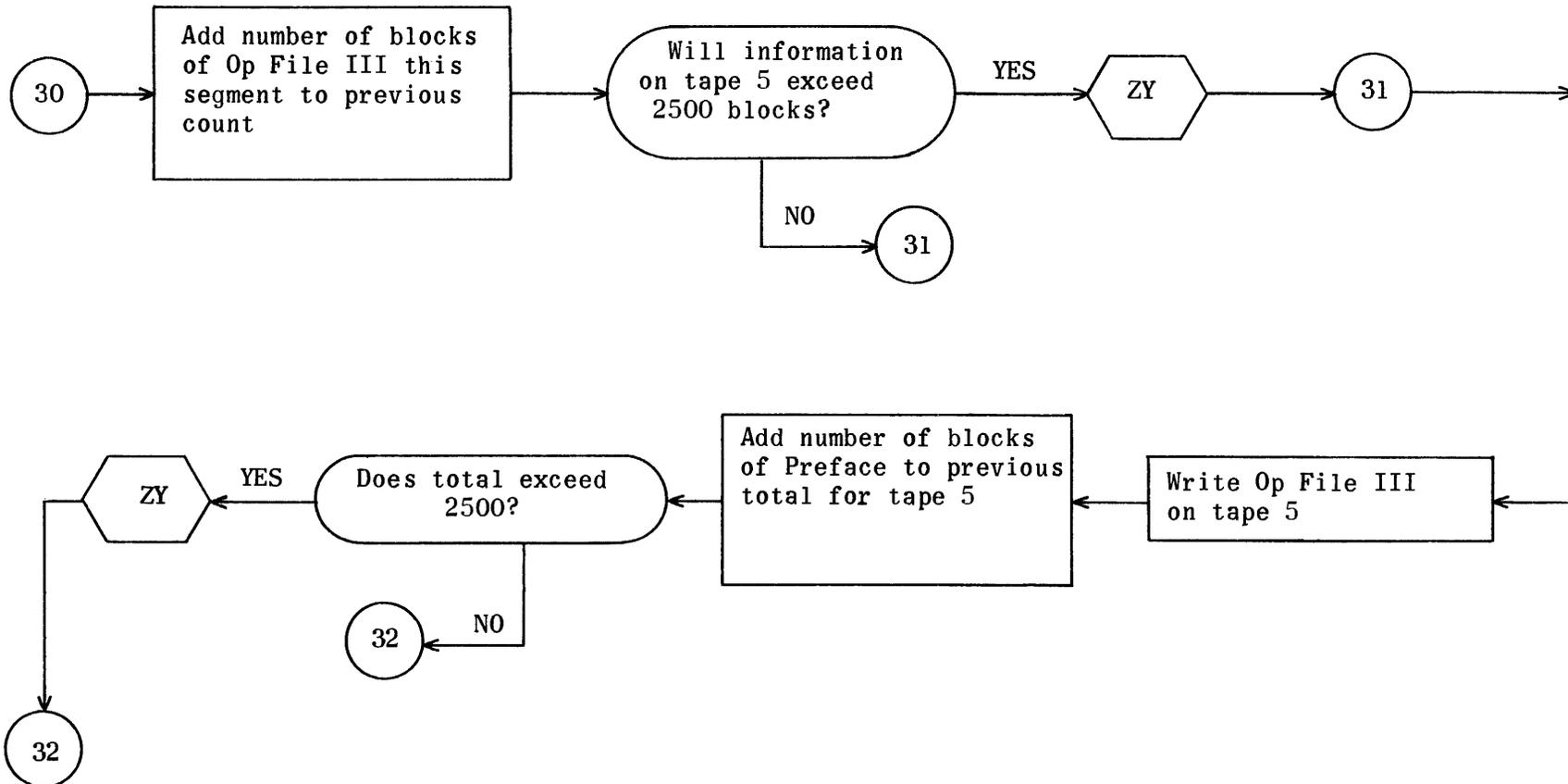




Allocation Flow Charts (Cont.)

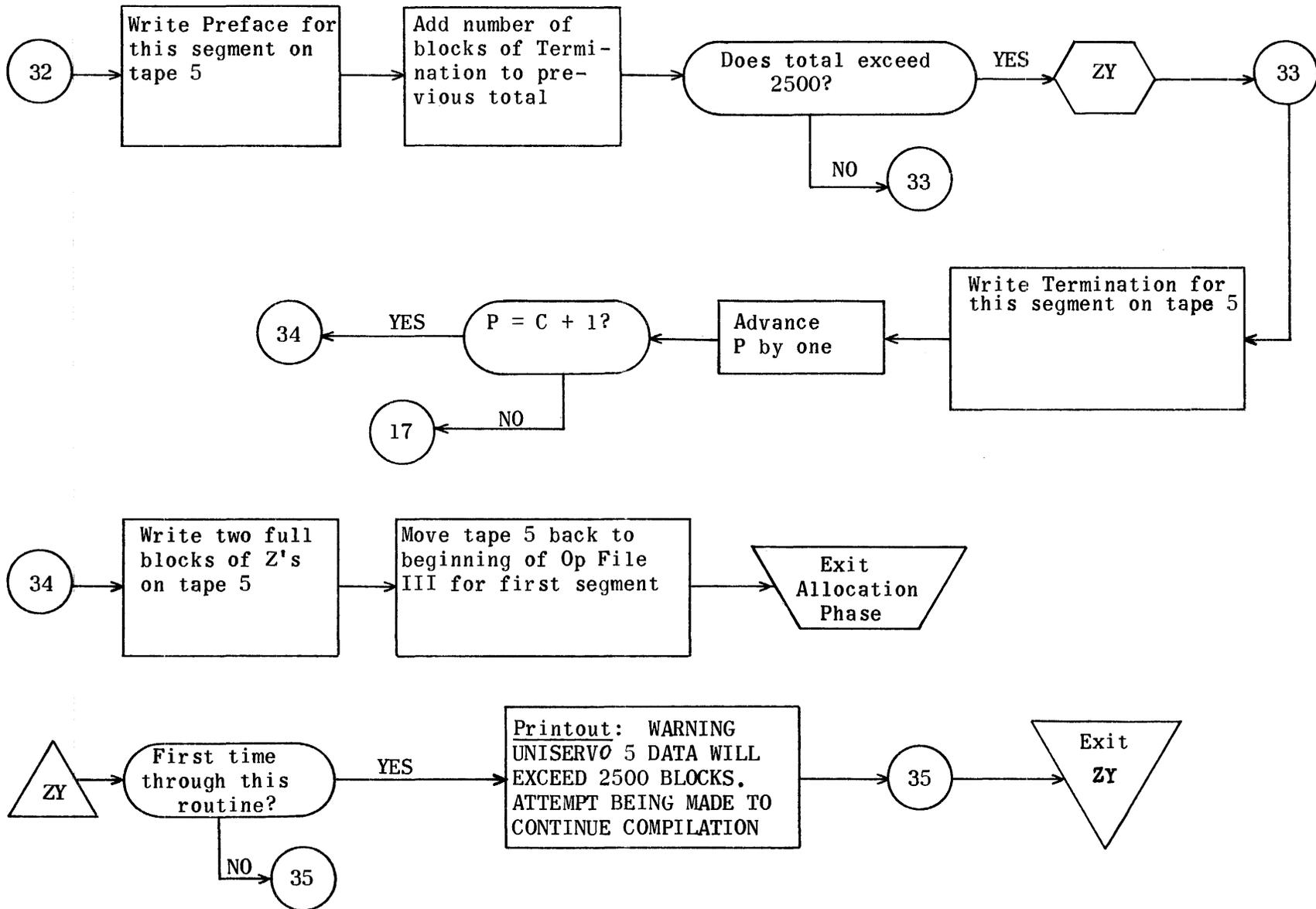


Allocation Flow Chart (Cont.)



1580

Allocation Flow Charts (Cont)



Allocation Regions

RE	GT21	}	Subroutines
RE	UP421		
RE	BR537		
RE	BQ632		
RE	CA674		
RE	CB763		
RE	CC1020		
RE	CD1056		
RE	CE1120		
RE	CF1175		
RE	CG1237		
RE	CH1266		
RE	CI1321		
RE	CJ1355		
RE	CK1407		
RE	CL1435	Stores information for Preface and Term.	
RE	CM1450	}	Build Preface and Term. in
RE	CN1505		buffer areas
RE	CP1522		Sets up "W" of RP-TP for exit of Term.
RE	BK1545		Preparation for writing onto tape
RE	CQ1557		Write Op File III onto tape
RE	CR1645		Write Preface, this segment onto tape
RE	CS1707		Write Termination this seg. onto tape
RE	CT1742		Exit region
RE	ZZ1760		Storage and constants
RE	ZY2144		Error Printout
RE	ZW2174		Warning Printout
RE	C02213		Patch correction regions (27) ₈ loc.
RE	FA3142	}	H.S.S. Address Op. F. 2A-6
RE	ZA3142		
RE	ZB2545		Fixed address of Directory 3
RE	ZC42102		Fixed drum address of Op File III
RE	ZD2644		Fixed address of Directory 4
RE	LD2242		Limit of drum (77000) ₈
RE	TL2243		Limit of tape (4704) ₈
RE	TI2355		Tape Image
RE	CU6		For assigning loc. for CT13 & CT16
RE	ZF7000		Fixed address for building Preface
RE	ZG7400		Fixed address for building Term.
RE	ZX76000		H.S.S. dump of TI for checkout
RE	BS76017		Region for generating M.S.'s in checkout
RE	ZE7230		Fixed address of LOC 2B
RE	CX2255	}	Patch correction allowing
RE	CZ2323		data arrays > 7777

Allocation

	IA	CA			
0	MJ	0	CO	} Read in Tape label (1st Bk)	
1	RJ	GT2	GT		i.e., 'FILE△△TWO△△△△'
2	TP	TI24	A	} Test for proper label	
3	EJ	ZZ43	CA5		of Tape #3 (1st Bk)
4	MJ	0	BR10		
5	TP	TI25	A		
6	EJ	ZZ44	CA10		
7	MJ	0	BR10		
10	TP	ZZ	CA66	Set connector A to A1	
11	TP	ZZ25	ZZ103	Segment #1→K	
12	TP	ZZ31	ZZ107	0→M (word count Directory 4)	
13	TP	ZZ31	ZZ62	0→Temp 1 (# lines rtne. for	
14	TP	ZZ16	ZZ106	current C/W)	
				MDAF3→G (fixed drum address	
15	TP	ZZ17	ZZ72	Op File III)	
16	TP	ZZ20	ZZ71	Set Dir. 4 & Dir. 3 to	
17	TP	ZZ31	ZZ105	fixed address	
①	20	TP	ZZ52	0→C (C _u = count of segments)	
	21	RJ	GT2	Read in next block	
	22	TP	TI	into TI	
	23	EJ	ZZ45	CA32	
	24	EJ	ZZ46	CA51	
	25	EJ	ZZ51	CA27	
	26	MJ	0	BR10	
	27	TP	TI24	A	
	30	EJ	ZZ51	CA65	
	31	MJ	0	BR10	
③⑧	32	TP	TI2	ZZ110	
	33	TP	TI4	ZZ113	
	34	TP	TI4	ZA2	
	35	TP	TI5	ZA3	
	36	TP	TI5	ZZ112	
	37	TV	ZZ21	ZZ102	
	40	SP	ZZ21	17	
	41	TU	A	CA44	
③⑨	42	TP	ZZ102	GT3	
	43	RJ	GT2	GT	
	44	TP	[30000]	A	
	45	EJ	ZZ47	CA20	
④③	46	RA	ZZ102	ZZ30	
	47	RA	CA44	ZZ27	
	50	MJ	0	CA42	
④④	51	RA	ZZ102	ZZ30	
	52	TP	TI2	ZZ111	
	53	TV	ZZ22	ZZ102	

	54	SP	ZZ22	17	}	Set up address for test of ENDΔ OF for Op File IIb
	55	TU	A	CA60		
(41)	56	TP	ZZ102	GT3	}	Read 1 block into IIb area
	57	RJ	GT2	GT		
	60	TP	30000	A	}	Test word for ENDΔ OF →CONN A
	61	EJ	ZZ47	CA66		
	62	RA	ZZ102	ZZ30	}	GENCOD + 120 → GENCOD Test word address + 120 → Test word address
	63	RA	CA60	ZZ27		
	64	MJ	0	CA56	}	Jump to read next block
(42)	65	TP	ZZ1	CA66		
(A)	66	[30	0	0]	}	Either MJ CONN A1 or CONN A2
		CA	CA67			
		IA	CB		}	Set up AJ & BI this segment LOC2A → AJ; LOC2B → BI
(A1)	0	TP	ZZ21	ZZ76		
	1	TP	ZZ22	ZZ77	}	Form test address to indicate end of Op File IIb list C + 1 → C (seg. counter) LOC2B → A _u
	2	SP	ZZ77	0		
	3	AT	ZZ111	ZZ101	}	Set commands with first address of Op File IIb
	4	MJ	0	CB5		
	5	RA	ZZ105	ZZ25	}	Is first word of Op File IIb = 0 ?
(1.5)	6	SP	ZZ22	17		
	7	TU	A	CB17	}	Zeroize 5th and 6th words of Op File IIa
	10	TU	A	CB33		
	11	TU	A	CB12	}	Jump to (3) Record first call word of Op File IIb into 5th word of Op File IIa and first word of Directory 4 Flag → next word of Op File IIa Insert segment # in second word of Directory 4 Dir. 4 + 2 → Dir. 4 (next loc.) M + 1 → M (count of items in Directory 4) Jump to (3) Dir. 4 → V
	12	SP	[30000]	0		
	13	ZJ	CB17	CB14	}	
	14	TP	A	ZA4		
	15	TP	A	ZA5	}	
	16	MJ	0	CC33		
	17	TP	[30000]	ZA4	}	
	20	MJ	0	CB27		
	21	TP	ZZ41	ZA5	}	
	22	SP	ZZ103	6		
	23	TP	A	30000	}	
	24	RA	ZZ72	ZZ32		
	25	RA	ZZ107	ZZ26	}	
	26	MJ	0	CC33		
	27	SP	ZZ72	0	}	
	30	TV	A	CB33		
	31	SA	ZZ26	0	}	
	32	TV	A	CB23		
	33	TP	30000	30000	}	
	34	MJ	0	CB21		
		CA	CB35			

②	0	IA	CC		
	1	SP	ZZ110	17	} Set n of RP to L; j = 2.
	2	SA	ZZ135	0	
	3	TU	A	CC6	
	4	SP	ZZ77	17	} Pick up "u" portion of BI in A
	5	TU	A	CC5	
	6	SP	[30000]	0	} Test Op File IIa for CW from Op File IIb
	7	RP	[30000]	CC10	
	10	EJ	ZA4	CC33	
	11	TP	A	ZZ115	Hold A_R in WS1
	12	SP	ZZ21	0	} Setup CC15
	13	SA	ZZ110	0	
	14	TV	A	CC15	
	15	TV	ZZ72	CC16	
		15	TP	ZZ115	[30000]
	16	TP	ZZ115	[30000]	
	17	SP	CC15	0	} Setup CC22
	20	SA	ZZ26	0	
	21	TV	A	CC22	
	22	TP	ZZ41	[30000]	Insert flag at Op.F. 2A + L + 1
	23	SP	CC16	0	} Set up CC27
	24	SA	ZZ26	0	
	25	TV	A	CC27	
	26	SP	ZZ103	6	
	27	TP	A	[30000]	Insert $K \times 2^{21}$ (seg. #) in Directory 4
	30	RA	ZZ72	ZZ32	Dir. 4 + 2 → Dir. 4 (next loc. available)
	31	RA	ZZ110	ZZ32	L + 2 → L (length + 2)
	32	RA	ZZ107	ZZ26	M + 1 → M (count of items in Directory 4)
③	33	RA	ZZ77	ZZ26	BI + 1 → BI (address next Op File IIb item)
	34	EJ	ZZ101	CD	Test for completion of BI test
	35	MJ	0	CC	Jump to CONN 2
		CA	CC36		
	0	IA	CD		
	0	RA	ZZ112	ZZ26	Set R to address following IP command
	1	TP	ZZ2	Q	Set up mask V
	2	QT	12	ZZ114	Mask '(S)' from location (12) ₈
	3	RA	ZZ113	ZZ114	D + S → D
	4	SP	ZZ76	0	} Form test address to indicate end of expanded IIa list
	5	AT	ZZ110	ZZ100	
④	6	SP	ZZ76	17	} CW address → A_u
	7	TU	A	CD16	
	10	SA	ZZ25	0	
	11	TU	A	ZZ115	l in u
	12	TU	A	CD15	(CW address) + 1 → (WS1) _u
	13	LT	25	A	Shift (CW address) + 1 to A_V

	14	TV	A	CD20	
	15	TV	[30000]	ZZ62	# lines in rtne in Temp 1
	16	SP	[30000]	0	CW→A
	17	LQ	ZZ62	17	Shift ^u # lines to u position
	20	TU	ZZ62	[30000]	# lines→u of second word of item
	21	LT	14	A	
	22	EJ	ZZ124	CD25	Test "26" CW
	23	EJ	ZZ125	CD25	Test "27" CW
	24	MJ	0	CE	Jump to ⑤
	25	SP	ZZ115	0	
	26	TU	A	CD27	
	27	SP	[30000]	0	
	30	TP	ZZ5	Q	Test for flag "14"
	31	QT	A	A	
	32	EJ	ZZ41	CE16	Jump→6
⑧	33	SP	ZZ115	0	(CW address) + 1→A _u
	34	LT	25	A	(CW address) + 1→A _v
	35	TV	A	CD36	
	36	TV	ZZ114	[30000]	Send S to v portion of 2nd word of item
	37	LQ	ZZ62	25	
	40	RA	ZZ114	ZZ62	
	41	MJ	0	CE16	Jump to ⑥
		CA	CD42		
		IA	CE		
⑤	0	EJ	ZZ126	CF	Test 77 type CW→⑨
	1	MJ	0	CE10	Assume 25, 24, 22, 5, or 4
	2	0	0	170	
	3	0	0	0	
	4	0	0	0	
	5	0	0	0	
	6	0	0	0	
	7	0	0	0	
	10	SP	ZZ115	0	(CW address) + 1→A _u
	11	LT	25	A	(CW address) + 1→A _v
	12	TV	A	CE13	
	13	TV	ZZ112	[30000]	R→v portion
	14	LQ	ZZ62	25	# lines shifted in Temp 1
	15	RA	ZZ112	ZZ62	R + # lines→R
⑥	16	RA	ZZ76	ZZ32	Address of CW address + 2
	17	EJ	ZZ100	CE21	Jump to ⑦ when end of Op File IIa reached.
	20	MJ	0	CD6	Jump ④
⑦	21	TP	ZZ103	FA	Seg. #→Op File III
	22	RA	ZZ110	ZZ32	L + 2→L
	23	SA	ZZ35	0	A = L; A + 4→A
	24	SA	ZZ106	0	Add G (next open M.D. address for Op File III)
	25	TJ	LD	CE27	Test limit of drum
	26	MJ	0	ZW1	→Error print & jump to BQ6 (rewind tape, etc.)
	27	SP	ZZ110	17	L→A _u

30	TP	A	ZA1	$A_u \rightarrow \# \text{ words Op File III}$	
31	SA	ZZ34	0	Add 4 in u	
32	SA	ZZ37	0	Add 3 in j	
33	TU	A	CE35		
34	TV	ZZ106	CE36		
35	RP	[30000]	CE37	} Transfer L + 4 words of	
36	TP	FA	[30000]		Op File III to MD at G
37	TV	ZZ71	CE42		
40	TV	ZZ71	CE44		
41	SP	ZZ106	17	$G \rightarrow A_u$	
42	TU	A	[30000]	$A_u \rightarrow \text{Directory 3}$	
43	MJ	0	CE52	$\# \text{ lines in Op File III} \rightarrow A_v$	
44	TV	A	[30000]	$A_v \rightarrow \text{Directory 3}$	
45	RA	ZZ71	ZZ26		
46	RA	ZZ103	ZZ25		
47	RA	ZZ106	ZZ110		
50	AT	ZZ35	ZZ106		
51	MJ	0	CA20	Jump to ①; read in next seg.	
52	SP	FA1	0		
53	LT	25	A		
54	MJ	0	CE44		
	CA	CE55			
	IA	CF			
⑨	0	SP	ZZ115	0	$(\text{CW address}) + 1 \rightarrow A_u$
	1	LT	25	A	$(\text{CW address}) + 1 \rightarrow A_v$
	2	TV	A	CF3	$D \rightarrow (\text{CW address}) + 1$
	3	TV	ZZ113	[30000]	} # lines + D \rightarrow D
	4	LQ	ZZ62	25	
	5	RA	ZZ113	ZZ62	} Jump to ⑥
	6	MJ	0	CE16	
①	7	SP	ZZ107	0	
	10	ZJ	CF13	CF11	
	11	TP	ZZ31	ZZ120	$0 \rightarrow \text{Index 1}$
	12	MJ	0	CH24	Jump to ①⑥
⑩ \rightarrow	13	TP	ZZ25	ZZ104	$l_u \rightarrow P_u$
	14	ST	ZZ26	ZZ121	$M - 1 \rightarrow \text{Index 2}$
⑪ \rightarrow	15	TP	ZZ17	ZZ72	Set Dir. 4 to fixed address
	16	SP	ZZ107	0	} $M - 1 \rightarrow \text{Index 1}$
	17	ST	ZZ26	ZZ120	
	20	SP	ZZ20	17	Add P
	21	SA	ZZ104	0	Subtract l_u Transfer u-portion
	22	SS	ZZ25	0	of p th word in Directory 3
	23	TU	A	CF25	to transfer command
	24	TU	A	CF26	and
	25	TU	[30000]	CF34	Transfer v-portion of p th word
	26	SP	[30000]	17	in Directory 3 to n of RP command
	27	MJ	0	CF35	+ 4 in u
	30	SA	ZZ34	0	+ 3 in j
	31	SA	ZZ37	0	
	32	TU	A	CF33	

33	RP	[30000]	CG	}
34	TP	[30000]	FA	}
35	TU	A	CG12	}
36	RA	CG12	ZZ36	}
37	TU	CF26	CF40	}
40	SP	30000	17	}
41	MJ	0	CF30	}
	CA	CF42		
	IA	CG		
⑫	0	TP	ZZ5	Q
	1	SP	ZZ72	17
	2	TU	A	CG11
	3	SA	ZZ25	0
	4	TU	A	CG5
	5	SP	[30000]	0
	6	QT	A	A
	7	EJ	ZZ41	CH
10	MJ	0	CG11	
11	SP	30000	0	
12	RP	30000	CH	}
13	EJ	ZA4	CG14	}
14	SP	CG12	0	
15	LQ	Q	17	
16	SS	Q	0	
17	SA	ZZ7	0	
20	TU	A	CG23	
21	TU	A	CH10	
22	TP	ZZ5	Q	}
23	SP	[30000]	0	}
24	QT	A	A	}
25	EJ	ZZ41	CH	}
26	MJ	0	CH5	}
	CA	CG27		

Transfer Op File III for
this seg. from M.D. to H.S.S.
Set up j n at CG12

Set up j n at CF33

Mask Op	→ Q	}	Pick up Op por- tion of word given by address at Dir. 4 + 1
Dir. 4	→ A _u		
Dir. 4 + 1	→ A		
Test Op	portion for '14' flag		
Set j	of RP to 2		
Obtain CW	given by address in Dir. 4		
Test this segment	Op File III		
for this CW			
j n	→ A _u		
j n - r	→ Q _v		
j n - (j n - r)	= + r → A _u		
Add fixed address	of Op File III		

Test for flagged CW in segment P

Jump to ⑬
Jump to ⑮

⑬	0	IA	CH	
		IJ	ZZ120	CH3
⑭	1	RA	ZZ104	ZZ25
	2	MJ	0	CF15
	3	RA	ZZ72	ZZ32
	4	MJ	0	CG
⑮	5	SP	ZZ72	0
	6	SA	ZZ26	0
	7	TV	A	CH10
10	TV	[30000]	[30000]	
11	TV	CH10	CH13	
12	TP	ZZ3	Q	
13	QS	ZZ104	[30000]	
14	TV	CH13	CH16	
15	TP	ZZ5	Q	
16	QS	ZZ41	[30000]	

Test that all Directory 4 checked
against this segment

P + 1_u → P
Jump to → ⑪
Dir. 4 + 2 → Dir. 4
Jump to → ⑫

Set v-portion CH₁₀ to address
given by Dir. 4 + 1
Record running address for this
CW at Dir. 4 + 1

Mask	→ Q	}	Insert "14" in Op of address given by ad- dress at Dir. 4 + 1
Record P x 2 ¹⁵	in Dir. 4 (segment to)		
Mask Op			

	17	IJ	ZZ121	CH	Test index 2 that all items Directory 4 processed
	20	MJ	0	CH21	
	21	TP	ZZ17	ZZ72	Set Dir. 4 to fixed address
	22	TP	ZZ107	ZZ161	Set Index l_A to M
	23	MJ	0	CH24	
①⑥	24	TP	ZZ25	ZZ104	$l_u \rightarrow P$, set to 1
	25	TP	ZZ10	ZZ63	
	26	TP	ZZ133	ZZ64	
	27	TP	ZZ134	ZZ65	Set Temps 1, 3, 4, 5 & 6 to Dummy instructions
	30	TP	ZZ133	ZZ66	
	31	TP	ZZ134	ZZ67	
	32	MJ	0	CI	Jump to \rightarrow ①⑦
		CA	CH33		
		IA	CI		
①⑦	0	SP	ZZ20	17	LOCD3 $\rightarrow A_u$
	1	SA	ZZ104	0	$A + P \rightarrow A$; Transfer u-portion of P th word of Directory 3 to the u-portion of the transfer command in CI16 and set n of the RP commands at CI15
	2	SS	ZZ25	0	$A_u = \#$ words in Op File III this seg.
	3	TU	A	CI5	Set # words for 77--- data search
	4	TU	A	CI6	Save # words in working Temp
	5	TU	[30000]	CI16	
	6	SP	[30000]	17	
	7	TU	A	CK7	
	10	TU	A	ZZ117	
	11	TU	A	CJ5	
	12	SA	ZZ34	0	+ 4
	13	SA	ZZ37	0	+ j = 3
	14	TU	A	CI15	
	15	RP	[30000]	CI32	Build Op File III image
	16	TP	[30000]	FA	
①⑧	17	TP	ZZ31	ZZ103	$0 \rightarrow K$
	20	IJ	ZZ161	CI22	Index l_A set initially to M
	21	MJ	0	CJ	Jump to CONN 19
	22	SP	ZZ72	17	Dir. 4 $\rightarrow A_u$
	23	TU	A	CJ4	Set address of Directory 4 item plus one
	24	SA	ZZ25	0	Set address of word 2 of Directory 4 item
	25	TU	A	CI30	
	26	TU	A	CJ24	
	27	TP	ZZ4	Q	Mask 'segment from' number into K (26-21)
	30	QT	[30000]	ZZ103	
	31	MJ	0	CJ	Jump to ①⑨
	32	RA	CJ5	ZZ36	+ j = 2
	33	MJ	0	CI17	
		CA	CI34		

19	0	IA	CJ			
	0	SP	ZZ104	6	P→A	
	1	EJ	ZZ103	CJ3	If P = K, go to 20 (00P00 = 00K00)	
	2	MJ	0	CJ27		
20	3	TP	C022	ZZ162	Reset value to 171 for region CM	
	4	SP	[30000]	0	Jump call word to A	
	5	RP	[30000]	CJ7	Locate this call word in Op File III	
	6	EJ	ZA4	CJ10	for segment P	
	7	MJ	0	BR6	Alarm 6	
	10	SP	CJ5	0	$j_n \rightarrow (A_R)_u$	
	11	LQ	Q	17	$j_n - r \rightarrow Q_u$	
	12	SS	Q	0	$r \rightarrow (A_R)_u$	
	13	SA	ZZ7	0	Finds address of 2nd word Op File III	
					item	
	14	TU	A	CJ16		
	15	TP	ZZ5	Q	Test if CW flagged in this segment.	
	16	SP	[30000]	0		
	17	QT	A	A		
	20	EJ	ZZ41	CJ22		
	21	MJ	0	BR6	Alarm 6	
	22	LQ	CJ16	Q25		
	23	TV	Q	CJ24	Replace Op File III word by second	
					word of Directory 4 item.	
	24	TP	[30000]	30000		
	25	RA	ZZ72	ZZ32	Dir. 4 + 2 → Dir. 4	
	26	MJ	0	CI17	Jump to 18	
	27	RA	FA5	ZZ26	Add 1 to H.S.S. of first "IP"	
	30	RA	ZZ161	ZZ26	Add 1 to index 1_A	
	31	MJ	0	CK	Jump to 21	
		CA	CJ32			
		IA	CK			
21	0	TP	ZZ11	ZZ74	Initialize Alpha and Beta	
	1	TP	ZZ12	ZZ75	(next address in Preface or Term.)	
	2	RA	CK7	ZZ36		
	3	SP	FA2	0		
	4	AT	12	ZZ113	Set "D" for Preface area	
	5	TP	A	ZZ141	and Temp D = # words = (S+R+2)+L () _R	
22	6	SP	ZZ153	0	76777 → A_u	
	7	RP	[30000]	CK24	→	search for
	10	TJ	[FA4]	CK11		data CW
	11	SN	Q	17	$- j_n + r \rightarrow A_u$	
	12	SA	CK7	0	+ j_n	calculate #
	13	TU	A	ZZ115	$r \rightarrow WS_1$	repeats
	14	RS	CK7	ZZ115	Set to continue search	
	15	RA	CK10	ZZ115		
	16	TU	A	CL		
	17	TU	A	CK22		
	20	RS	CK22	ZZ25	Test if above TJ command	
	21	SP	ZZ41	0	reacted on a "14" in the Op code	
	22	TJ	30000	CK6		
	23	MJ	0	CL		

29

24	TU	ZZ7	CK10
25	MJ	0	CP
	CA	CK26	

Reset (TJ FA4 CK11) on exit

Stores Information Necessary In Building Termination and Preface

24	0	IA	CL			
	0	SP	[30000]	0	2nd word of 77--- data item→A	
	1	MJ	0	CX2		
27	2	TV	A	ZZ137	H.S.S. address →Temp B _v	
	3	RS	CL	ZZ25		
	4	TU	CL	CL5		
	5	SP	[30000]	0	CW→A _u	
	6	TP	ZZ6	Q	L(00777) _u →Q	
	7	QT	A	A	Mask and multiply by 2	
	10	LA	A	1		
	11	AT	ZZ143	ZZ140	MD address of array→Temp C _u	
	12	MJ	0	CM		
		CA	CL13			
		IA	CM			
	0	TU	ZZ136	ZZ64	Set up RP command for Preface	
	1	RA	ZZ64	ZZ37		
	2	TU	ZZ64	ZZ66	Set up RP command for Term.	
	3	TV	ZZ137	ZZ65	Set data H.S.S. address for Preface	
	4	SP	ZZ140	0		
	5	TU	A	CM6	Set up address of array on MD	
	6	TU	[30000]	ZZ65		
	7	TU	A	CM10		
	10	SP	[30000]	0	Set up address of array on MD	
	11	LT	25	A	for Term.	
	12	TV	A	ZZ67		
	13	SP	ZZ137	17	Set data H.S.S. address for Term.	
	14	TU	A	ZZ67		
	15	SP	ZZ74	0	Calculate # words in Preface	
	16	SS	ZZ11	0		
	17	TJ	ZZ162	CM23	Test # words < 170	
	20	RA	ZZ162	ZZ146	Increment by 170	
	21	MJ	0	CM22		
	22	TP	C021	ZZ147		
	23	RJ	CM23	CM31	1 shot switch	
	24	TV	ZZ113	ZZ64		
	25	RA	ZZ113	ZZ32	D + 2 → D	} W of RP
	26	RA	ZZ147	ZZ32		
	27	TV	ZZ147	ZZ66	TE + 2 → TE	
	30	MJ	0	CN		
	31	SP	ZZ37	0		
	32	LT	25	A	Send (30000) _v to first Preface	
					RP _w command	
	33	TV	A	ZZ64		
	34	MJ	0	CM26		
		CA	CM35			

	IA	CN		}	Set up transfer commands (i.e., fill in v-addresses)
0	TV	ZZ74	CN6		
1	RA	ZZ74	ZZ26		
2	TV	ZZ74	CN7	}	Transfer RP - TP setup to proper location in buffer area
3	TV	ZZ75	CN10		
4	RA	ZZ75	ZZ26		
5	TV	ZZ75	CN11	}	Update available locations in buffer area
6	TP	ZZ64	[30000]		
7	TP	ZZ65	[30000]		
10	TP	ZZ66	[30000]	}	Jump to continue searching list
11	TP	ZZ67	[30000]		
12	RA	ZZ74	ZZ26		
13	RA	ZZ75	ZZ26		
14	MJ	0	CK6		
	CA	CN15			

Setup "W" of RP - Commands for Exit of Termination

0	IA	CP		
0	TV	ZZ152	CM23	Reset 1 shot switch
1	RS	ZZ75	ZZ32	} Insert "CT16" in W of last RP command in Termination
2	TV	ZZ75	CP3	
3	TV	ZZ151	[30000]	
4	SP	ZZ74	0	
				ALPHA → A } # of entries in Pref- ace area → A
5	ST	ZZ11	A	
6	TJ	CE2	BK	Test $A \leq 170$.
7	DV	ZZ146	ZZ120	i.e. $A > 170$ indicates more than
10	MJ	0	CP11	one block needed. Dividing
11	ZJ	CP13	CP12	# of entries by $(170)_8$ gives
12	RS	ZZ120	ZZ26	# of blocks needed. $A = 0$
13	SP	ZZ12	0	indicates an integral
14	SA	ZZ163	0	number of blocks needed.
15	TV	A	CP20	$A = \#$ of blocks needed -1
16	IJ	ZZ120	CP20	if $Q \neq 0$ or # blocks if $Q = 0$.
17	MJ	0	BK	Set index 1 to A and let
20	TV	ZZ150	[30000]	the index control the number
21	RA	CP20	ZZ146	of times (CT13) is inserted
22	MJ	0	CP16	
	CA	CP23		

Preparation for Writing Onto Tape

0	IA	BK		
0	RJ	BK	BK2	
1	MJ	0	CQ	
2	TP	14	Q	} Sum number of blocks already written
3	QT	ZZ2	ZZ157	
4	LQ	Q	17	
5	QT	BK11	A	
6	AT	ZZ157	ZZ157	} Save # blocks already written
7	TP	ZZ157	ZZ160	
10	MJ	0	CQ	
11	0	0	77	
	CA	BK12		

Write Op File III Onto Tape #5

	IA	CQ		
0	RP	10170	CQ2	} Fill TI with Z's
1	TP	ZZ51	TI	
(30) 2	RA	ZZ157	ZZ26	} Test for exceeding tape length
3	TJ	TL	CQ5	
4	RJ	ZY	ZY1	} F I L E Δ 3
(31) 5	TP	ZZ154	TI	
6	TP	ZZ155	TI1	S E G Δ Δ Δ
7	RP	30004	CQ11	} Read first 4 words
10	TP	ZA	TI2	
11	SP	ZZ74	0	} of Op File III image → Tape image
12	SS	ZZ11	17	
13	TP	A	TI6	(ALPHA-LOCPRE) = # words
14	LT	25	ZZ142	} Calculate # words in Preface
15	TP	ZZ53	GT3	
16	RJ	GT2	GT	→ save in (TEMPE) _v
17	LQ	ZZ117	25	} Write first block on tape
20	TP	ZZ2	Q	
21	QT	ZZ117	A	} Calculate # words in Op File III
22	DV	ZZ30	ZZ121	
				this seg.
23	LT	10017	ZZ112	} Record # full blocks required into
24	TU	ZZ7	CQ33	
25	IJ	ZZ121	CQ27	Index 2
26	MJ	0	CQ37	Shift remainder → R _u
27	RA	ZZ157	ZZ26	LOC Op File (3)
30	TJ	TL	CQ32	} Have all full blocks been written?
31	RJ	ZY	ZY1	
32	RP	30170	CQ34	→ Jump → (34)
33	TP	[30000]	TI	} Test for exceeding tape length
34	RJ	GT2	GT	
35	RA	CQ33	ZZ27	} Transfer 120 words from File III
36	MJ	0	CQ25	
37	MJ	0	CQ62	image into TI
40	TJ	TL	CQ42	Write 1 full block on tape
41	RJ	ZY	ZY1	Advance Op File III image address
42	RP	10170	CQ44	→ Jump → (33)
43	TP	ZZ51	TI	} Test for exceeding length
44	TU	ZZ112	CQ47	
45	RA	CQ47	ZZ37	of tape
46	TU	CQ33	CQ50	Fill TI with Z's
47	RP	[30000]	CQ51	} Set N of RP command
50	TP	[30000]	TI	
51	RJ	GT2	GT	3 in j
52	RA	ZZ157	ZZ26	Set "u" of transfer command
53	TJ	TL	CQ55	} Transfer partial block to TI
54	RJ	ZY	ZY1	
55	RP	10170	CQ57	} Write partial block
56	TP	ZZ51	TI	
				Test exceeding length of tape
				Fill with Z's

57	TP	ZZ47	TI	E N D O F
60	RJ	GT2	GT	
61	MJ	0	CR	
62	TP	ZZ112	A	} Handles special case where 0 mod 170 words are written.
63	ZJ	CQ64	CQ52	
64	RA	ZZ157	ZZ26	
65	MJ	0	CQ40	
	CA	CQ66		

Write Preface for This Seg. Onto Tape #5

	IA	CR		
0	RP	10170	CR2	} Fill TI with Z's
1	TP	ZZ51	TI	
2	SP	ZZ142	0	# words in Preface → A _u
3	ZJ	CR36	CT	# full blocks
4	TP	Q	ZZ122	→ Index 1 ; TEMPT
5	LT	10017	ZZ112	# words in partial block
6	SP	ZZ11	17	Set up u of transfer command
7	TU	A	CR16	
10	IJ	ZZ120	CR12	Have all full blocks been written?
11	MJ	0	CR40	
12	RA	ZZ157	ZZ26	
13	TJ	TL	CR15	} Test for exceeding length of tape
14	RJ	ZY	ZY1	
15	RP	30170	CR17	Transfer 1 full block
16	TP	[30000]	TI	into TI
17	RJ	GT2	GT	Write 1 full block onto tape #5
20	RA	CR16	ZZ27	Advance u-address by (120) ₁₀
21	MJ	0	CR10	
22	RA	ZZ157	ZZ26	
23	TJ	TL	CR25	} Test exceeding block length
24	RJ	ZY	ZY1	
25	RP	10170	CR27	Fill TI with Z's
26	TP	ZZ51	TI	
27	TU	ZZ112	CR32	Set up RP command to
30	RA	CR32	ZZ37	# of words in partial block
31	TU	CR16	CR33	Set up TP command
32	RP	[30000]	CR34	Read partial block into TI
33	TP	[30000]	TI	
34	RJ	GT2	GT	Write 1 block onto tape #5
35	MJ	0	CS	
36	DV	ZZ30	ZZ120	
37	MJ	0	CR4	
40	TP	ZZ112	A	
41	ZJ	CR22	CS	
	CA	CR42		

(32)

Write Termination For This Seg. Onto Tape #5

33

0	IA	CS		
	TP	ZZ122	ZZ120	Set up Index 1 = # full blocks to be written
1	SP	ZZ12	17	} Set up "u" of TP command
2	TU	A	CS11	
3	IJ	ZZ120	CS5	
4	MJ	0	CS31	Jump to write partial block
5	RA	ZZ157	ZZ26	} Test for exceeding length of tape
6	TJ	TL	CS10	
7	RJ	ZY	ZY1	
10	RP	30170	CS12	} Read 1 block into TI
11	TP	[30000]	TI	
12	RJ	GT2	GT	} Write 1 block onto tape #5
13	RA	CS11	ZZ27	
14	MJ	0	CS3	} Increase address of TP command
15	RA	ZZ157	ZZ26	
16	TJ	TL	CS20	} Test for exceeding length of block
17	RJ	ZY	ZY1	
20	RP	10170	CS22	} Fill TI with Z's
21	TP	ZZ51	TI	
22	TU	ZZ112	CS25	} Setup RP - TP commands
23	RA	CS25	ZZ37	
24	TU	CS11	CS26	
25	RP	[30000]	CS27	} Read partial block into TI
26	TP	[30000]	TI	
27	RJ	GT2	GT	} Write 1 block
30	MJ	0	CT	
31	TP	ZZ112	A	} Test for 0 mod 170 entries
32	ZJ	CS15	CT	
	CA	CS33		
0	IA	CT		
	RA	ZZ104	ZZ25	} Advance P by 1
1	SP	ZZ105	0	
2	SA	ZZ25	0	} C + 1 → A
3	EJ	ZZ104	C06	
4	MJ	0	C04	Then → CT5
5	SP	ZZ157	0	Then jump to 17
6	SS	ZZ160	25	
7	TP	A	CT14	
10	RA	CT15	CT14	
11	TP	CT15	GT3	
12	RJ	GT2	GT	
13	MJ	0	77010	Exit allocation phase
14	0	0	0	Parameter for repositioning tape
15	40	5	0	
	CA	CT16		

34

	IA	CO		
0	RA	ZZ52	20	} Add TN to code word for tape handler
1	RA	ZZ102	20	
2	TP	ZZ52	GT3	
3	MJ	0	CA1	} Reset initialization value for termination, then → 17
4	TP	C021	ZZ147	
5	MJ	0	CI	} Write double block of Z's
6	RA	ZZ157	ZZ26	
7	TJ	TL	C011	
10	RJ	ZY	ZY1	
11	RP	10170	C013	
12	TP	ZZ51	TI	
13	RJ	GT2	GT	
14	TV	C020	C011	
15	MJ	0	C06	
16	RJ	GT2	GT	
17	MJ	0	CT5	} Mask for counting blocks written on tape
20	0	0	C016	
21	0	0	610	
22	0	0	171	

CA C023

	IA	ZW		
0	MJ	0	BQ6	P R O B L E M Δ T O O Δ L O N G . Δ Δ D R U M Δ S T O R A G E Δ E X C E E D E D Δ B Y Δ A L L O C A T I O N Δ F I L E .
1	TP	ZW16	UP3	
2	RJ	UP2	UP	
3	MJ	0	ZW	
4	52	54512	54630	
5	47	01665	15101	
6	46	51503	22201	
7	01	27546	74701	
10	65	66515	42432	
11	30	01307	22630	
12	30	27302	70125	
13	73	01244	64651	
14	26	24663	45150	
15	01	31344	63022	
16	0	ZW4	12	
	CA	ZW17		

Print Error Warning of Exceeding Length of Tape

35

	IA	ZY		
0	MJ	0	[30000]	
1	RJ	ZY1	ZY3	One shot switch
2	MJ	0	ZY	
3	TP	ZY27	UP3	
4	RJ	UP2	UP	
5	MJ	0	ZY	
6	71	24545	03450	W A R N I N
7	32	22010	16750	G . Δ Δ U N
10	34	65305	47051	I S E R V O
11	01	10012	72466	Δ 5 Δ D A T
12	24	01713	44646	A Δ W I L L
13	01	30722	63030	Δ E X C E E
14	27	01051	00303	D Δ 2 5 O O
15	01	25465	12645	Δ B L O C K
16	65	22010	12466	S . Δ Δ A T
17	66	30475	26601	T E M P T Δ
20	25	30345	03201	B E I N G Δ
21	47	24273	00166	M A D E Δ T
22	51	01010	10101	O Δ Δ Δ Δ Δ
23	01	01265	15066	Δ Δ C O N T
24	34	50673	00126	I N U E Δ C
25	51	47523	44624	O M P I L A
26	66	34515	02277	T I O N . 77
27	0	ZY6	21	
	CA	ZY30		

	IA	CX		
0	TU	A	ZZ136	} Exit to main program
1	MJ	0	CL2	
2	TJ	CZ	CX	} Test # lines > 7777
3	TP	A	CZ6	
4	RS	CL	ZZ25	} Save information
5	TU	CL	CX6	
6	SP	30000	0	} Compute address which
7	TP	ZZ6	Q	
10	QT	A	A	} contains address where S.S.
11	LA	A	1	
12	AT	ZZ143	ZZ140	} data is stored on drum
13	TV	CZ6	ZZ137	
14	TU	CZ3	ZZ136	} Core address of beginning of array
15	RJ	CN14	CM	
16	RS	CZ6	CZ3	} # words set to 7777
17	TJ	CZ	CX27	
20	TJ	CZ2	CX22	} Build Preface and Term.
21	MJ	0	CX22	
22	TU	CZ3	ZZ136	} Reduce number of words by 7777
23	RA	ZZ137	CZ4	
24	RJ	CX35	CX36	} 1 core < # lines ≤ 2 cores
25	RJ	CN14	CM	
26	RS	CZ6	CZ3	} 2 core < # lines ≤ 3 cores
27	TU	CZ6	ZZ136	
30	LQ	CZ6	25	} Update MD address
31	RA	ZZ137	CZ4	
32	RJ	CX35	CX36	} Process Preface and Term.
33	TP	CX45	CN14	
34	MJ	0	CM	} Update H.S.S. address
35	MJ	0	30000	
36	TU	ZZ140	CX37	} Update MD address
37	TU	30000	CX44	
40	RA	CX44	CZ3	} Reset Exit in main program
41	TU	CX43	ZZ140	
42	MJ	0	CX35	} Routine for updating MD address
43	0	CX44	0	
44	0	0	0	} Save information
45	MJ	0	CK6	
	CA	CX46		

(25)

(26)

	IA	CZ		
0	0	10000	0	
1	0	17777	0	
2	0	27776	0	
3	0	7777	0	
4	0	0	7777	
5	0	17776	0	
6	0	0	0	Temp
	CA	CZ7		

		IA	ZZ							
	0	MJ	0	CB	MJ	0	CONN A1			
	1	MJ	0	CF7	MJ	0	CONN A2			
	2	0	0	77777						
	3	00	00077	0	Jump "to" mask					
	4	0	7700	0	Jump "from" mask					
	5	77	0	0						
	6	0	00777	0						
	7	0	ZA4	0	Fixed address of start of Op File III					
LOC77	10	0	0	0	Fixed drum address for starting 77 data					
LOCPRE	11	0	0	ZF	Fixed address for Preface					
LOCTER	12	0	0	ZG	Fixed address for Termination					
LOCD5	13	0	0	0						
LOCFCA	14	0	0	0						
LOCBA	15	0	0	0						
DLOCF3	16	0	0	ZC	Fixed drum address for Op File III					
LOCD4	17	0	0	ZD	Fixed address of Directory 4					
LOCD3	20	0	0	ZB	Fixed address of Directory 3					
LOC2A	21	0	0	ZA6	Fixed address Op File IIa (H.S.S.)					
LOC2B	22	0	0	ZE	Fixed address (by segment) Op File IIb					
LOCCSA	23	0	0	0						
LOCCTA	24	0	0	0						
	25	0	1	0						
	26	0	0	1						
	27	0	170	0						
	30	0	0	170						
ZERO	31	0	0	0						
	32	0	0	2						
	33	0	2	0						
	34	0	4	0						
	35	0	0	4						
	36	0	20000	0						
	37	0	30000	0						
CONST	40	0	0	0						
FLAG	41	14	0	0						
DATA	42	0	0	0						
	43	31	34463	00101	F	I	L	E	Δ	Δ
	44	66	71510	10101	T	W	0	Δ	Δ	Δ
	45	66	71510	12401	T	W	0	Δ	A	Δ
	46	66	71510	12501	T	W	0	Δ	B	Δ
	47	30	50270	15131	E	N	D	Δ	0	F
	50	01	30506	65473						
	51	74	74747	47474	Z	Z	Z	Z	Z	Z
	52	50	00103	TI	GT code word to read Op File IIa					
	53	71	00105	TI						
	54	0	0	0						
	55	0	0	0						
	56	0	0	0						
N	57	0	0	[30000]	Length of control (also initial address of S)					
MINUS	60	0	0	0						

ENDBUF	61	0	0	0	
TEMP1	62	0	0	0	
2	63	0	0	0	
3	64	0	0	0	
4	65	0	0	0	
5	66	0	0	0	
6	67	0	0	0	
7	70	0	0	0	
	71	0	0	0	Dir. 3 - next open address of Directory 3
	72	0	0	0	Dir. 4 - next open address of Directory 4
	73	0	0	0	Not used
ALPHA	74	0	0	0	Next open address in Preface
BETA	75	0	0	0	Next open address in Termination
AJ	76	0	0	0	Address next Op File IIa item.
BI	77	0	0	0	Address next Op File IIb item.
AJTEST	100	0	0	0	
BITEST	101	0	0	0	
GENCOD	102	50	00103	[30000]	
K	103	0	0	0	
P	104	0	0	0	
C	105	0	0	0	
G	106	0	0	0	
M	107	0	0	0	Count of Directory 4 items
L	110	0	0	0	
1	111	0	0	0	
R	112	0	0	0	Next address to assign to routines
D	113	0	0	0	Next address to assign data
S	114	0	0	0	
WS1	115	0	0	0	
WS2	116	0	0	0	
WS3	117	0	0	0	
INDEX1	120	0	0	0	
INDEX2	121	0	0	0	
TEMPT	122	0	0	0	
	123	0	0	3	
	124	0	0	26	
	125	0	0	27	
	126	0	0	77	
	127	0	0	25	
	130	0	0	24	
	131	0	0	22	
	132	0	0	5	
	133	75	0	0	
	134	11	0	0	
	135	0	20002	0	
TEMPA	136	0	0	0	
TEMPB	137	0	0	0	

TEMPC	140	0	0	0	
TEMPD	141	0	0	0	
TEMPE	142	0	0	0	
	143	0	40101	0	
	144	0	0	165	
	145	0	0	167	
	146	0	0	170	
TE	147	0	0	610	
CT13	150	0	0	CU13	
CT16	151	0	0	CU16	
	152	0	0	CM31	For resetting 1 shot switch
	153	0	76777	0	
	154	31	34463	00106	F I L E Δ 3
	155	01	01653	03201	Δ Δ S E G Δ
	156	0	0	TL	
TEMPBKCTR1	157	0	0	0	
TEMPBKCTR	160	0	0	0	# blocks already written
INDEX1A	161	0	0	0	
	162	0	0	170	
	163	0	0	166	
		CA	ZZ164		

3. INITIALIZATION GENERATOR

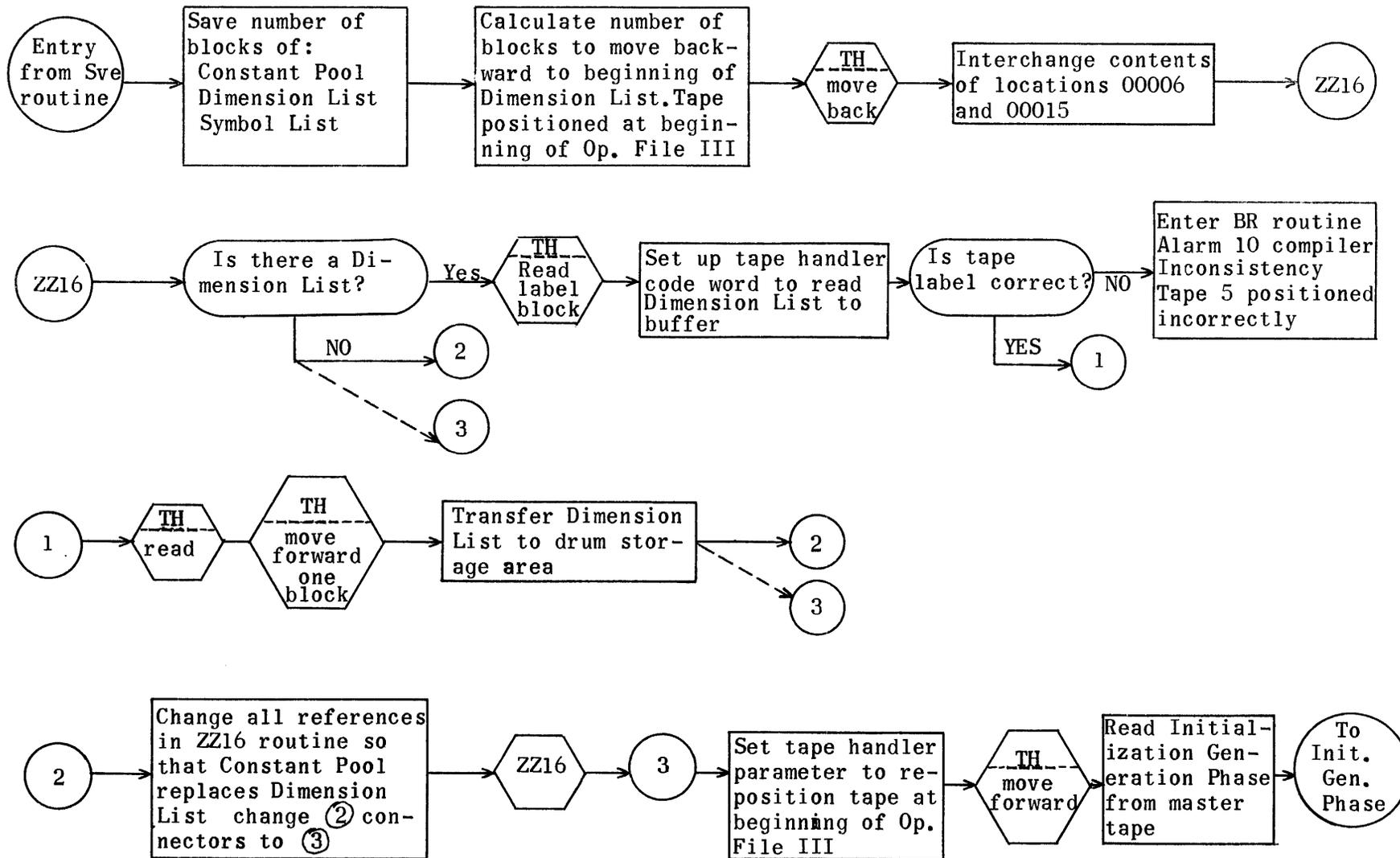
3. Initialization Generator

Initialization Generation Setup

The Setup Routine for Initialization Generation reads the original Dimension List and the Constant Pool from magnetic tape and stores them on the drum. These lists do not overlay the modified Dimension List that was built by the Allocator Setup Routine since it will be used by the Processing Phase later.

The counters at locations 00006 and 00015 are interchanged so that 00006 becomes the Dimension List counter for this phase.

After reading the Dimension List and Constant Pool the tape is repositioned to the beginning of Op File III. The 14 blocks of the Initialization Generation Phase are then read from the UNICODE master tape and control is transferred to it.



Initialization Generation Set-Up Flow Chart

Regions for Initialization Generation Setup

RE	ZZ7230	(37)
RE	ZW7267	(15)
RE	ZX7304	(14)
RE	ZT7320	(3)
RE	TH21	Tape handler
RE	BR537	Compiler Inconsistency Routine
RE	DL42102	(6000) Dimension List
RE	CL50102	(1000) Constant Pool
RE	ZY700	Buffer
RE	IG2000	Loading and entry address for IG
RE	IL1600	168 = # blocks of Initialization Generation phase

Initialization Generation Setup Routine

	IA	ZZ		
0	SP	14	0	
1	LT	3	ZT	Save # blocks of Constant Pool
2	SP	A	0	
3	LT	6	ZT1	Save # blocks of Dimension List
4	SP	A	0	
5	LT	6	ZT2	Save # blocks of Symbol List
6	SP	ZT	0	
7	SA	ZT1	0	
10	SA	ZT2	25	} Calculate and move # blocks backward to beginning of Dimension List (if any)
11	AT	ZX7	TH3	
12	RJ	TH2	TH	
13	TP	15	Q	
14	TP	6	15	} Interchange contents of 6 and 15
15	TP	Q	6	
16	SP	[ZT1]	0	} Is there a Dimension list (Constant Pool)?
17	ZJ	ZZ20	[ZW]	
20	TP	ZX2	TH3	} Read label block to H.S.S.
21	RJ	TH2	TH	
22	SP	[ZT1]	0	
23	SS	ZX	25	} Set up code word to read Dimension List (Constant Pool)
24	AT	ZX3	TH3	
25	TP	ZY	A	} Check label
26	EJ	[ZX10]	ZZ30	
27	MJ	0	BR12	
30	RJ	TH2	TH	} Read Dimension List (Constant Pool) to H.S.S.
31	TP	ZX4	TH3	
32	RJ	TH2	TH	} Move past end label
33	TU	[6]	ZZ35	
34	RA	ZZ35	ZX1	} Set up transfer
35	RP	30000	[ZW]	
36	TP	ZY	[DL]	} Transfer Dimension List (Constant Pool) to storage area
	CA	ZZ37		

①

2

	IA	ZW	
0	TU	ZZ6	ZZ16
1	TU	ZZ6	ZZ22
2	TV	ZX12	ZZ17
3	TU	ZX12	ZZ26
4	TU	ZX13	ZZ33
5	TV	ZX13	ZZ36
6	RJ	ZZ35	ZZ16
7	SP	ZT2	25
10	AT	ZX5	TH3
11	RJ	TH2	TH
12	TP	ZX6	TH3
13	RJ	TH2	TH
14	MJ	0	IG
	CA	ZW15	

Set for reading Constant Pool

Go to read Constant Pool and transfer
Move forward past Symbol List

Read Initialization Generation
to H.S.S.
Jump into Phase

3

	IA	ZX	
0	0	0	2
1	0	10000	0
2	50	105	ZY
3	50	5	ZY
4	30	105	0
5	30	5	0
6	50	IL1	IG
7	40	5	0
10	27	34473	05065
11	26	51506	56624
12	0	ZX11	ZW7
13	0	10	CL
	CA	ZX14	

Read one block of Uniservo 5
General Read for Uniservo 5
Move one block forward
General move forward
IL = (# of blocks of Initialization
Generation) * 100
General move backward - tape 5
D I M E N S
C O N S T A

Initialization

It is convenient to divide this write-up into two sections. The first describes the initialization phase proper, and the second explains the actual generation. The distinction between the two should be kept in mind at all times.

Running Initialization

There are two classes of operation that may be considered;

(1) functions always performed, and (2) functions whose operation depends upon the circumstances of the particular object program compiled. We may tabulate these two classes as follows:

Functions always performed	Functions sometimes performed
<ol style="list-style-type: none">1) Rewind program tape2) Clear 1 core bank to zero3) Load GTH coding4) Load Control coding5) Load Constant Pool <p style="text-align: center;">↓</p> <p>and finally, transfer control to Control coding, to pull in Segment 1</p>	<ol style="list-style-type: none">1) Read in and translate "DATA INDEX" from either paper or magnetic tape.2) Determine which data tapes are required by the program, and check that these are mounted in Uniservos.3) Load values for a certain class of subscripted variables to their appropriate drum area.

All coding, constants, etc., necessary for all parts of Initialization are written on the Object program tape preceding Segment 1 of the generated coding. See Page 1617, Layout of Object Program Tape.

The coding is entered to H.S.S., and control transferred there at appropriate points, by means of the Object Program Loader Routine, which is part of the UNICODE Service Library (Sect. II, 1, c, (2).) The loader performs items 1 and 2 in the list above of "functions always performed". From this point onward, the loader merely loads data, and transfers control to such data as indicated. In other words, computer operations are entirely guided by what is present on Object Program Magnetic Tape #1.

In all problems, the first operation at this point is to load the Generalized Tape Handler into the operating locations it will occupy throughout running of the Object Program. After this, the procedure will vary, depending on the program under consideration. We may discuss the case where all possibilities are included, for the sake of completeness, and reference to the diagram on page 1617 should make clear which portions are variable.

From this point initialization may be divided into three sections. The first two are optional; the third invariant.

Section 1.

This coding is required in one form or another, if, in the compiled program, there are any input subscripted variables. These are defined as subscripted variables either referenced by Read sentences, or appearing only on the right-hand side of equations. This coding causes a "DATA INDEX" to be read in, either from paper tape, or from magnetic tape on Uniservo 2, translates it to the form required by the running program, and then performs certain checking operations.

Because Read sentences specify only variables to be read in, and not the Uniservo on which they are to be found, as in the original UNICODE program, the DATA INDEX is necessary to supply this information. It informs the

computer of the tape on which a given variable is written, its position relative to other data, and its identifying tape label. This permits data to be referenced in the program by one symbol, and labelled on tape by a different name.

After the index has been set up, the next operation scans the index, checking that all variables required by input operations at various stages of running are present. While this is being done, a list is built of the Uniservo numbers containing the data required, and the next operation in sequence rewinds all these servos, and checks that they do, in fact, hold data tapes. Note that checking does not go so far as to check the contents of these data tapes. This would be excessively time-consuming, and if a tape is found at a later stage which does not contain data that the index says it should, the machine will stop, and a correct tape may then be substituted.

A further operation of this section is to build a list of all input variables (see previous definition) not specifically referenced by Read sentences. This list, called List B, is for use by Section 2 of Initialization. Its format is shown on page 1618.

When all these operations are satisfactorily completed, control is returned to the loader, which pulls in more tape from Uniservo 1, and acts according to the data thereon. This will probably be Section 2 of initialization, described below.

Section 2.

This section is needed whenever the "automatic data read-in" facility is utilized. It accepts as input LIST B, produced by the preceding section of Initialization, which contains the XS3 representations of the variables required, the number of values of each required (the modulus), and the drum addresses pertaining. The conditions under which subscripted variables are "automatically" read in are:

- 1) The variable should appear only on the right-hand side of equations.
- 2) The variable is not referenced by any Read sentence.

The number of values specified by the relevant Dimension statement are read in. If there are less than this on tape, the computer will indicate this and stop. On continuing, the remaining values are filled in with zeros.

The basic reading is performed by a subroutine essentially identical to the Read library subroutine. The annotated coding and flow charts for this subroutine begin on page 106 of this manual, and should be consulted for further information.

At the conclusion of operations, control is returned to the UNICODE Loader to pull in more tape and perform the remaining functions of Initialization.

Section 3.

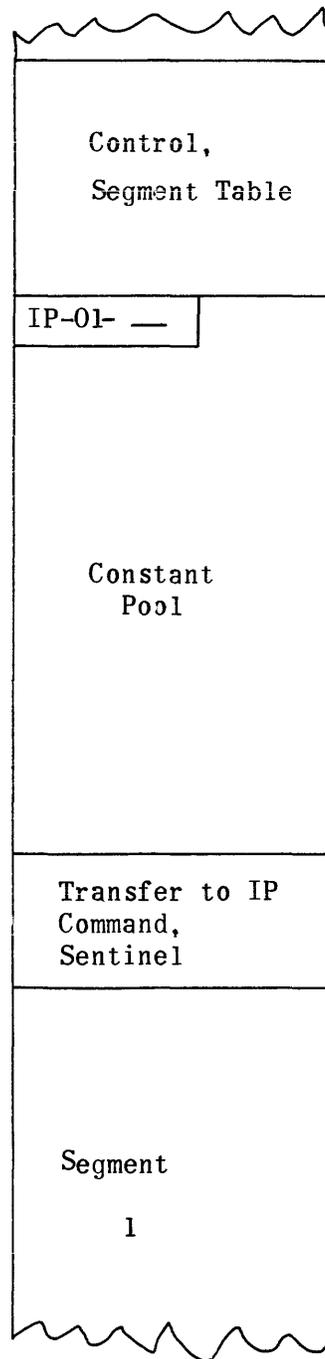
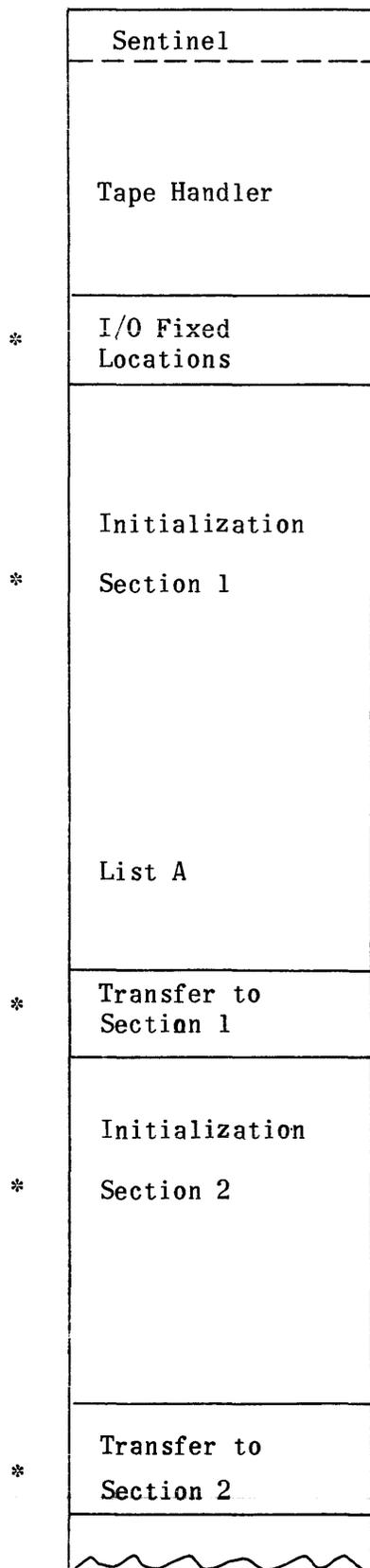
This includes all the remaining operations of Initialization, which are always performed, whatever the nature of the program compiled. They are a series of loading operations, followed by a transfer to an IP command which causes UNICODE Control to take over and initiate the running of Segment 1.

The material loaded is listed below in the order of loading.

- 1) UNICODE Control
- 2) Segment table
- 3) IP order
- 4) Constant Pool

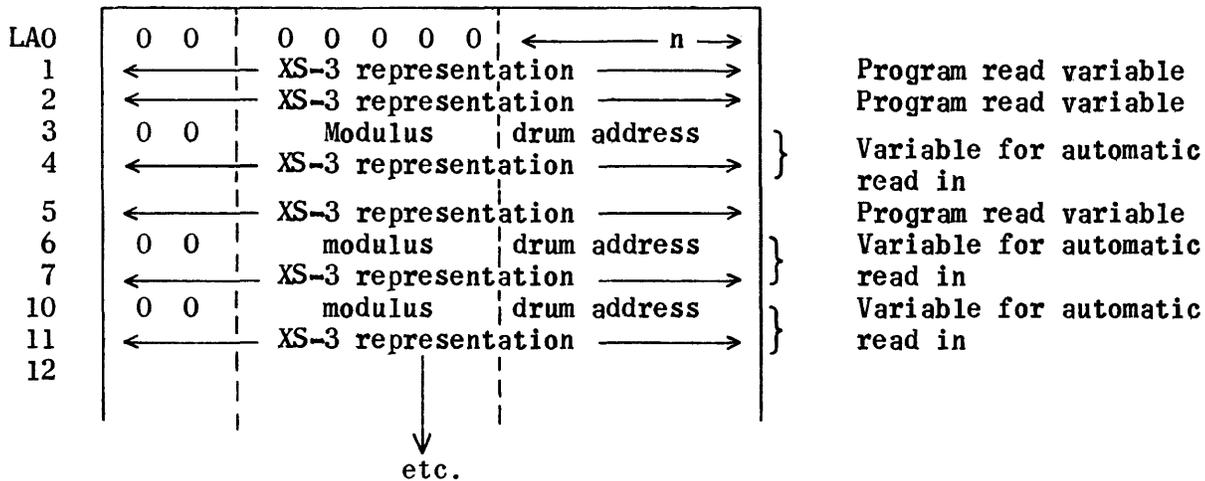
Initialization Generation.

The generation of Initialization takes place after the Allocation phase. The generation itself is relatively simple. First, the leading sentinels and the GTH are written on tape. Then some tests are made on the contents of the Dimension List to determine which, if any, variables are input variables. Depending on the results, values for the I/O fixed locations and such variable coding as may be necessary are written on tape, together with LIST A. Finally, Control is written, followed by the table of Segment lengths and the Constant Pool. The Constant Pool is preceded by an IP order, designed, when operative, to pull in Segment 1 and start running.



* Indicates those sections which may be absent

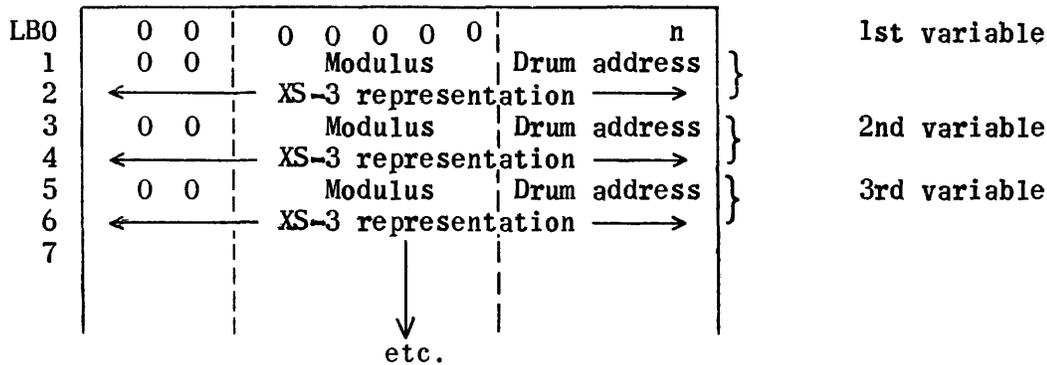
Layout of Object Program Tape



n = number of items in list.
(e.g., above case through LA11, n = 6)

List A format (built by Initialization Generation)

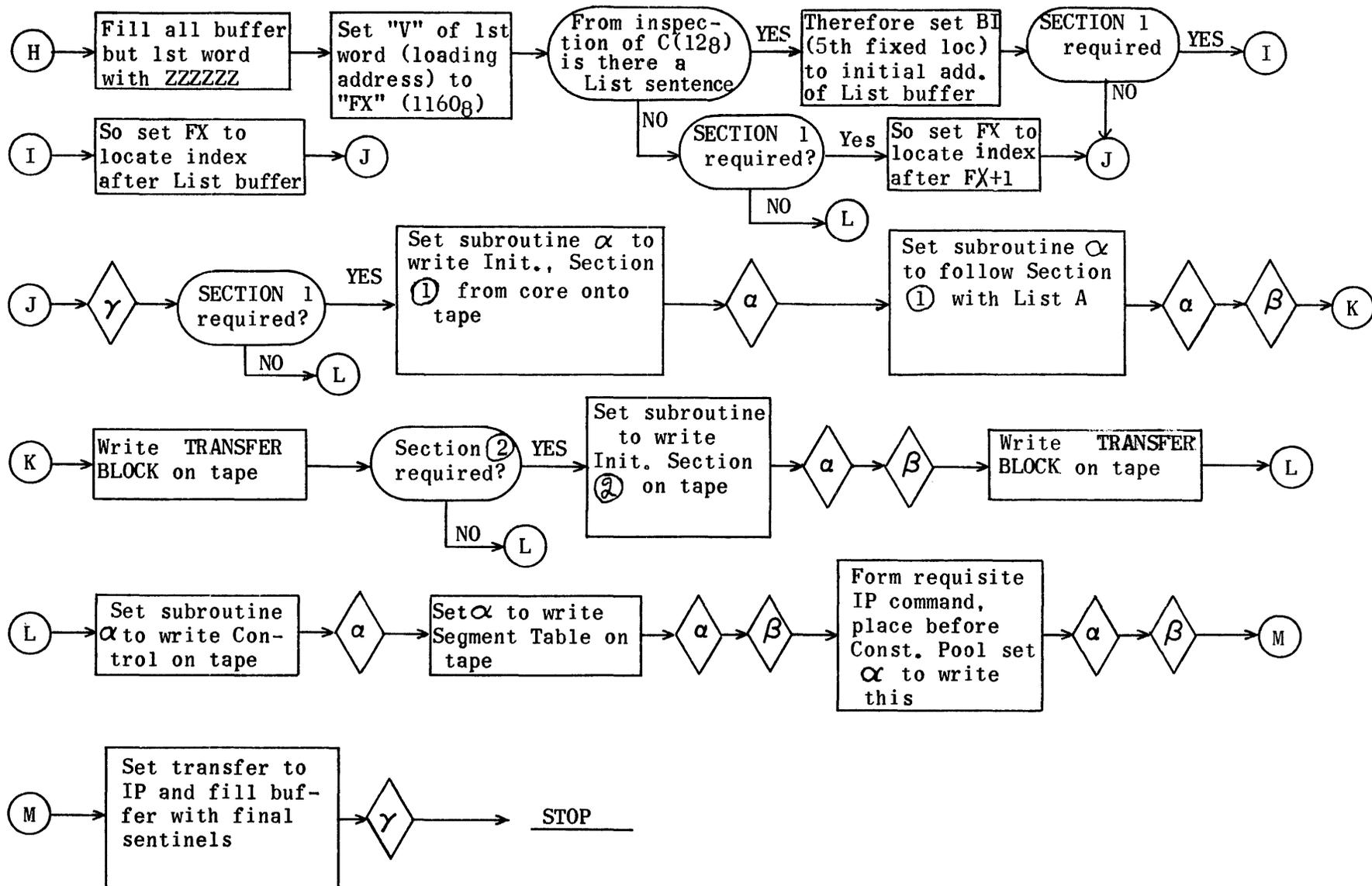
This list is derived from the Dimension List and includes all subscripted variables that are to be read in for a program.



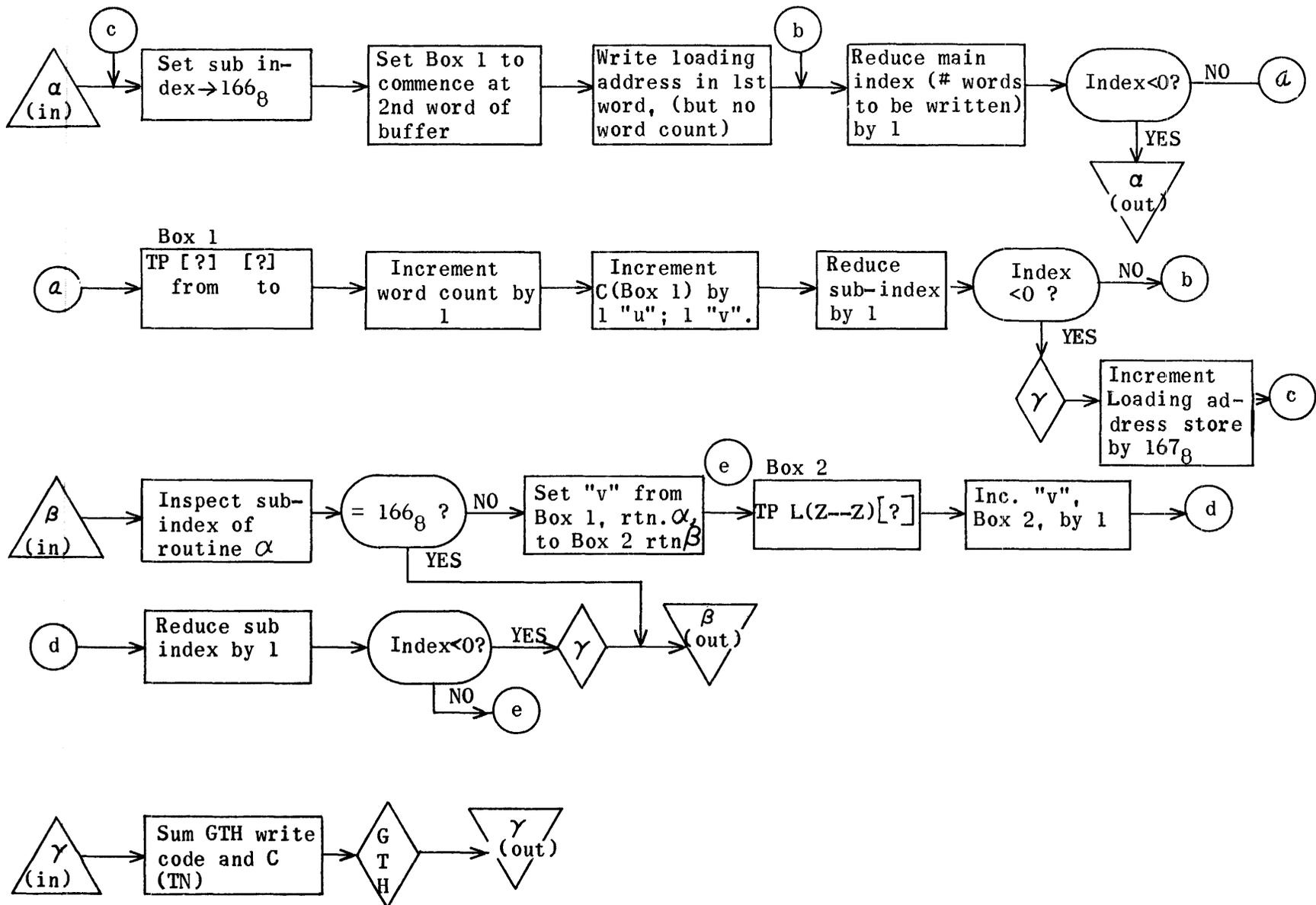
n = number of items in list
(e.g., above case through LB6, n = 3)

List B format (built by Initialization, Section 1)

This list is derived from List A and includes all variables to be automatically read.



Initialization Generation Flow Chart (Cont.)



Initialization Generation Flow Chart (Cont.)

Regions for Initialization Generation

RE GI2000	Loading address
RE IG2000	Coding for Initialization Generation
RE IA2354	Stored coding for Section 1 of Initialization
RE ID3460	Stored coding for Section 2 of Initialization
RE ON4274	Stored coding for Control Section
RE TG4461	Stored coding for Tape Handler
RE WB5300	Output buffer
RE FL5470	List A
RE DL42102	Dimension List
RE CL50102	Constant Pool
RE ST653	Segment Table
RE TN20	Indicator word for number of Uniservos
RE GH21	Tape handler for compilation
RE UP421	Uniprint Routine
RE BR537 } RE RB640 }	Compiler Inconsistency Routine
RE HD115	HD + 73 = operating address of Object Program Tape Handler
RE LG400	Length of Tape Handler
RE FX1000	Fixed I/O locations
RE LI1104	Length of Section 1
RE IN2000	Initial operating address of Section 1
RE LN614	Length of Section 2
RE DD1750	Initial operating address of Section 2
RE LC165	Length of Control Section
RE LT21	Length of Segment Table
RE ZA77000	Service Routine

Initialization Generation

IG	0	IA GI			
	1	TP IG304	A	} Rewind appropriate servo (3 or 6)	
	2	RJ IG250	IG246		
		2	TP IG327	WB	Set 1st word (loading address + W.C. for 3 bkts.)
		3	RP 10023	IG5	} Fill in rest of 1st bkt. with Z---Z
		4	TP IG251	WB1	
		5	RP 30004	IG7	} UNICODE Δ OBJECT Δ PROGRAM Δ
		6	TP IG252	WB24	
		7	RP 10044	IG11	} Fill in 2nd and 3rd bkts. with Z---Z
		10	TP IG251	WB30	
		11	TP IG330	IG230	Set index to length of (stored) GTH
		12	TP IG331	IG232	Set "Load Add. Temp" store (<u>no</u> word count)
		13	TU IG332	IG221	Initialize to start of (stored) GTH
		14	TP IG265	IG231	Set block index to 73 ₈
		15	TV IG307	IG221	Initialize to WB74
		16	RJ IG220	IG217	Go write GTH
		17	RJ IG242	IG233	Conclude any unfinished block
		20	TP IG302	IG353	Set indicator for "no index" (large +ve no.)
		21	TV IG126	IG145	Assume no "Automatic Read"
		22	TP IG273	WB	First CW is 77000
		23	TP IG262	WB1	} Set index to # of 77--- type CW's
		24	TV 6	WB1	
		25	TV IG310	IG107	Initialize List A building to start at FL1
		26	TP IG262	FL	Zeroize List A item counter
		27	TP IG263	IG112	1 → List A line counter
		30	TU 6	IG34	Set up RP
		31	IJ WB1	IG33	
		32	MJ 0	IG113	Exit to next section
		33	TP WB	A	CW for extraction → A
		34	RP[0]	BR1	Alarm if not present
		35	EJ DL	IG36	
		36	SN Q	17	
		37	SA IG34	0	
			CA GI40		
	IG	40	IA GI40		
		41	SA IG35	0	
		42	TU A	IG42	
43		TP [30000]	Q	"X" and mod. line → Q	
44		QT IG303	A	Inspect all "X"	
45		ZJ IG70	IG45	Auto-read required?	
46		TV IG311	IG145	Yes. So note (→ IG146)	
47		QT IG301	WB2	Extract modulus from u-field	
48		TJ IG300	IG62	Test with 2,501 ₁₀	
49		TU IG42	IG52	Too large an array.	
50	RS IG52	IG275	} Go back 2 lines for XS3 rep.		
51	TP [30000]	IG314			

	53	TP IG312	UP3	} Print-out.
	54	RJ UP2	UP	
	55	SP IG262	0	Clear A
	56	MS 0	IG57	Alarm stop
	57	ZJ RB	IG60	
	60	TP IG310	Q	Place 2,500 ₁₀ in Q "u"
	61	MJ 0	IG46	and back.
	62	TU IG42	IG64	} Go back 3 lines for drum address
	63	RS IG64	IG276	
	64	LQ [30000]	Q25	
	65	TV Q	WB2	Complete line for List A
	66	RJ IG111	IG106	Store it (<u>not</u> incrementing item count)
	67	MJ 0	IG72	
	70	QT IG302	A	"X" ≠ 0, so inspect "read-bit"
	71	ZJ IG72	IG77	If no prog. "read-out"?
	72	RS IG42	IG275	Prog. read. relates, so note the XS3 name
	73	TU A	IG74	(Go back 2 lines for XS3 name)
	74	TP [30000]	WB2	
	75	RJ IG111	IG101	Store in List A (Incrementing item count)
	76	TP IG262	IG353	Note index wanted (zeroize indicator register)
	77	RA WB	IG263	Prepare for next CW
		CA GI100		
		IA GI100		
IG	100	MJ 0	IG31	And then back.
	101	RA FL	IG263	Increment item count
	102	TJ IG264	IG106	Test with 51 ₁₀
	103	TP IG321	UP3	} List A building
	104	RJ UP2	UP	
	105	MJ 0	RB	
	106	RA IG112	IG263	
	107	TP WB2	[30000]	
	110	RA IG107	IG263	
	111	MJ 0	[30000]	
	112	[0 30000	30000]	
	113	RP 10167	IG115	
	114	TP IG251	WB1	
	115	TP IG333	WB	Fill block with Z---Z
				Set loading address (FX), but <u>no</u> word count
	116	TP 12	Q	
	117	QT IG274	A	} Is there a List order?
	120	ZJ IG121	IG126	
	121	TP IG334	WB5	Yes, set BI
	122	TU IG307	WB	And set word count → 5
	123	IJ IG353	IG131	Index wanted?
	124	TP IG335	WB1	Yes, set FX
	125	MJ 0	IG131	
	126	IJ IG353	IG155	No List order, but index wanted?
	127	TP IG336	WB1	Yes, set FX

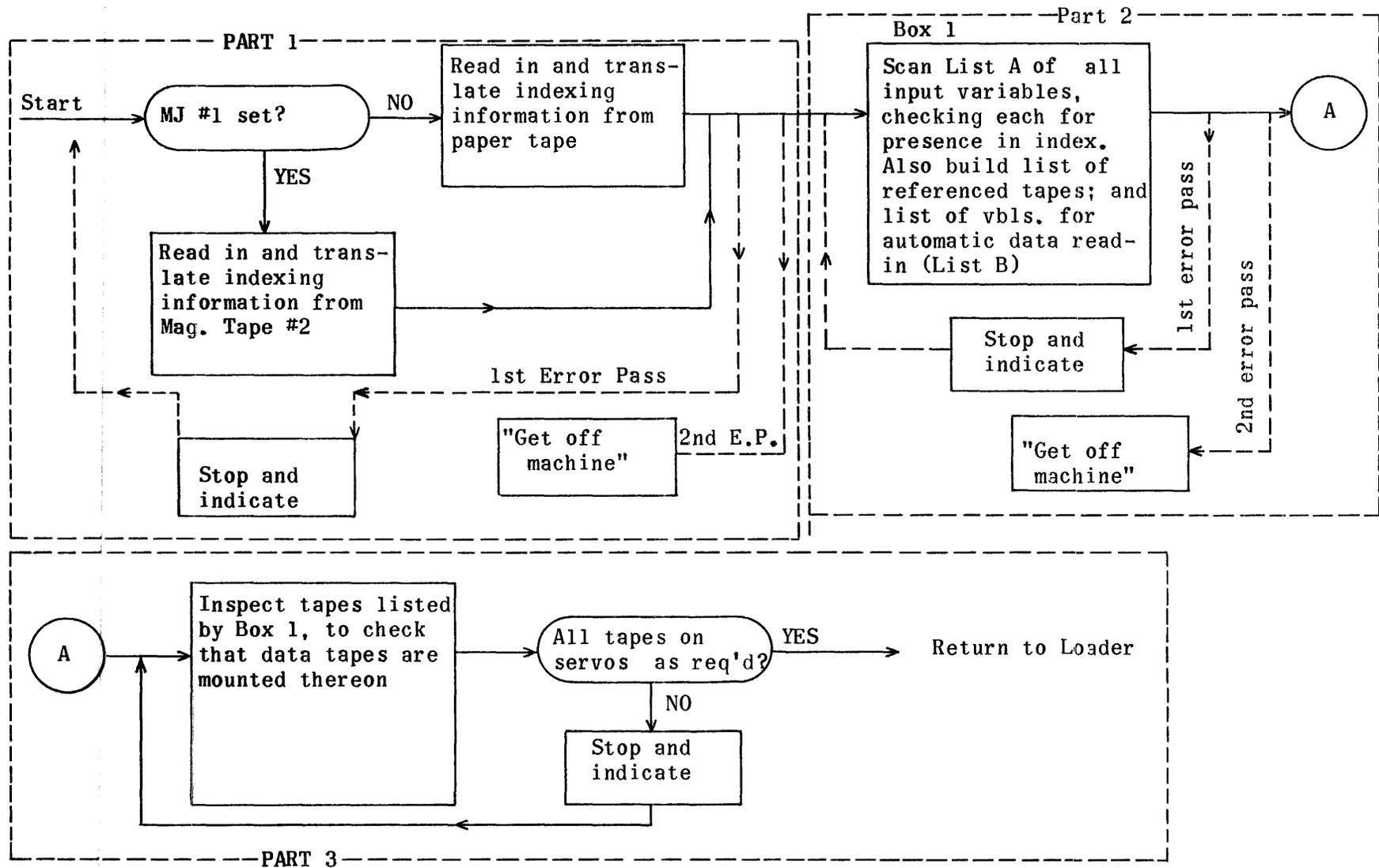
	130	TU IG274	WB	And set word count → 1
	131	RJ IG250	IG245	Now write block
	132	IJ IG353	IG155	Index wanted?
	133	TP IG337	IG230	Yes, set in length of Init. ①
	134	TP IG340	IG232	Set "Loading Add. Temp" store. (no word count)
	135	TU IG341	IG221	Initialize to where Init. ① stored.
	136	RJ IG220	IG214	Go write tape
	137	TP IG112	IG230	Now, add List A. Set index
		CA GI140		
		IA GI140		
IG	140	TU IG101	IG221	Pick it up from FL
	141	RJ IG220	IG217	Go write tape
	142	RJ IG242	IG233	Conclude any unfinished block
	143	TP IG340	WB	Set "transfer" word
	144	RJ IG250	IG243	And fill in rest of block
	145	MJ 0	[30000]	Automatic read wanted? (if not IG155)
	146	TP IG342	IG230	Yes. Set index to length of Init. ②
	147	TP IG343	IG232	Set "Loading Add. Temp" store
	150	TU IG344	IG221	Initialize to where Init. ② is stored.
	151	RJ IG220	IG214	Go write
	152	RJ IG242	IG233	Conclude any unfinished block
	153	TP IG343	WB	Set "transfer" word
	154	RJ IG250	IG243	And fill in rest of block
	155	TP IG345	IG230	Set index to length of Control (Excluding Seg. Tab)
	156	TP IG346	IG232	Set "Loading Add. Temp" store
	157	TU IG347	IG221	Initialize to where Control is stored.
	160	RJ IG220	IG214	Go write
	161	TP IG350	IG230	Set index for length of segment table
	162	TU IG351	IG221	Initialize to where ST is stored.
	163	RJ IG220	IG217	Go write ST
	164	RJ IG242	IG233	Conclude any unfinished block.
	165	TP 10	Q	Using v mask, note initial address of CP
	166	QT IG272	A	
	167	ST IG263	WB	Subtract 1 to leave room for IP
	170	TP A	IG232	Set "Loading Address Temp" store
	171	TP A	IG353	Save it for "transfer" word
	172	TU IG274	WB	Set W.C. → 1 (as at least IP)
	173	LQ 10	Q25	With partial v mask, set index to length of CP
	174	QT IG271	IG230	
	175	TU IG352	IG221	Initialize to where CP stored.
	176	TP IG306	WB1	Basic IP
	177	TV 12	WB1	
		CA GI200		
		IA GI200		
IG	200	RA WB1	IG263	Increment by 1 to complete IP
	201	TP IG266	IG231	Set block index to 165
	202	TV IG65	IG221	Initialize to WB2

	203	RJ IG220	IG217	Go write
	204	RJ IG242	IG233	Conclude any unfinished block
	205	TP IG353	WB	Set "transfer"
	206	RP 30004	IG210	END Δ OF Δ INIT
	207	TP IG256	WB1	
	210	RP 10163	IG212	Fill with Z----Z
	211	TP IG251	WB5	
	212	RJ IG250	IG245	Go write
	213	MJ 0	ZA10	End. Back to Service Routine.
Write	214	TP IG267	IG231	Set index → 166
①	215	TV IG4	IG221	Initialize to WB1
	216	TP IG232	WB	Write 1st word ([W.C.] l.a.)
	217	IJ IG230	IG221	Jump on main index
	220	MJ 0	[30000]	Exit
	221	TP [30000]	[30000]	Count 1 word.
	222	RA WB	IG274	
	223	RA IG221	IG277	Jump back on block index.
	224	IJ IG231	IG217	
	225	RJ IG250	IG245	Block full-go write it.
	226	RA IG232	IG270	Increment "Ld. Add. Temp." by 167 (V)
	227	MJ 0	IG214	Main index
	230	[0 0	0]	
	231	[0 0	0]	Block index
Write	232	[0 0	0]	"Loading address temp" store.
②	233	SP IG231	0	If block index = 166 ₈ , no
	234	EJ IG267	IG242	partial block to finish
	235	TV IG221	IG236	Fill with Z----Z
	236	TP IG251	[30000]	
	237	RA IG236	IG263	
		CA GI240		
		IA GI240		
IG	240	IJ IG231	IG236	Go write
	241	RJ IG250	IG245	Exit
	242	MJ 0	[30000]	Fill with Z---Z
	243	RP 10167	IG245	
Write	244	TP IG251	WB1	Go write on tape
③	245	TP IG305	A	
	246	AT TN	GH3	Exit
	247	RJ GH2	GH	
	250	MJ 0	[30000]	Z Z Z Z Z Z (XS3)
	251	74 74747	47474	U N I C O D } (XS3)
	252	67 50342	65127	
	253	30 01512	54430	
	254	26 66015	25451	C T Δ P R O } (XS3)
	255	32 54244	72201	
	256	30 50270	15131	
	257	01 34503	46634	E N D Δ O F } (XS3)
	260	24 46346	52466	
	261	34 51500	10101	
	262	0 0	0	Δ I N I T I } (XS3)
	263	0 0	1	
	264	0 0	63	
				I O N Δ Δ Δ

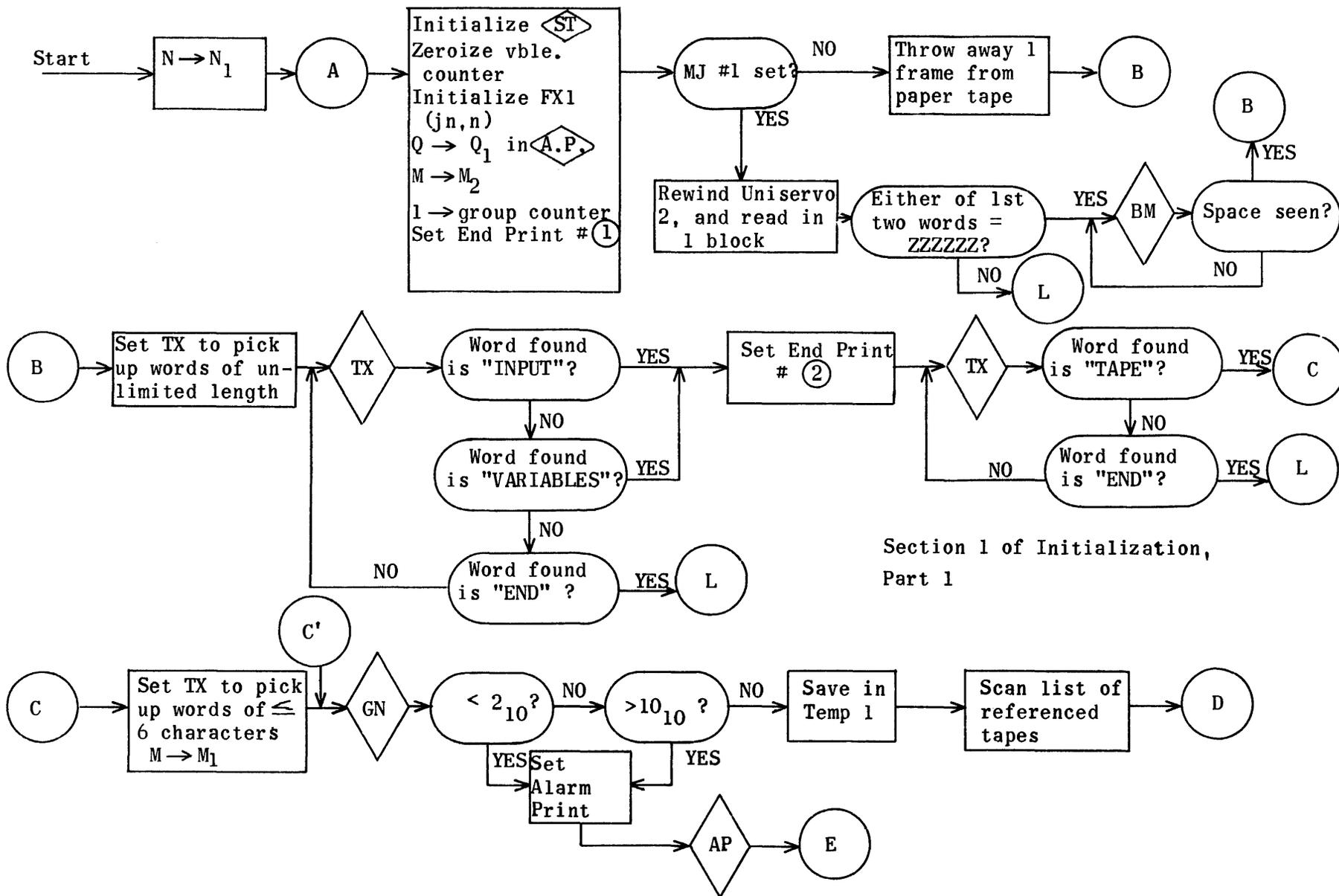
51₁₀

	265	0 0	73	
	266	0 0	165	
	267	0 0	166	
	270	0 0	167	
	271	0 0	07777	
	272	0 0	77777	
	273	0 0	77000	
	274	0 1	0	
	275	0 2	0	
	276	0 3	0	
	277	0 1	1	
		CA GI300		
		IA GI300		
IG	300	0 4705	0	2501 ₁₀
	301	0 77777	0	
	302	01 0	0	Read bit mask
	303	07 0	0	"X" mask
	304	10 3	0	Rewind Uniservo 3 code.
	305	71 00103	WB	Write Uniservo 3 code
	306	IP 00001	0	Basic IP
	307	0 5	WB74	
	310	0 4704	FL1	
	311	0 0	IG146	
	312	0 IG313	6	1st print-out parameter.
	313	24 54542	47301	A R R A Y Δ
	314	[77 77777	77777]	[Name]
	315	01 30722	63030	Δ E X C E E
	316	27 65010	51003	D S Δ 2 5 0
	317	03 01702	44667	0 Δ V A L U
	320	30 65227	77777	E S .
	321	0 IG322	5	
	322	66 51510	14724	T O O Δ M A
	323	50 73013	45052	N Y Δ I N P
	324	67 66012	45454	U T Δ A R R
	325	24 73652	20154	A Y S . Δ R
	326	30 71543	46630	E W R I T E
	327	0 73	HD	HD + 73 = 1st operating address of GTH. (= GT)
	330	0 0	LG	Length of GTH
	331	0 0	HD	
	332	0 TG	0	1st address of stored GTH.
	333	0 0	FX	
	334	0 0	FX5	BI setting
	335	0 FX175	FX175	FX Setting for index if List buffer present
	336	0 FX2	FX2	FX Setting for index if no list buffer present
	337	0 0	LI	length of Section ① , initialization
		CA GI340		
		IA GI340		
IG	340	0 0	IN	Initial running address of Section ①
	341	0 IA	0	Stored here

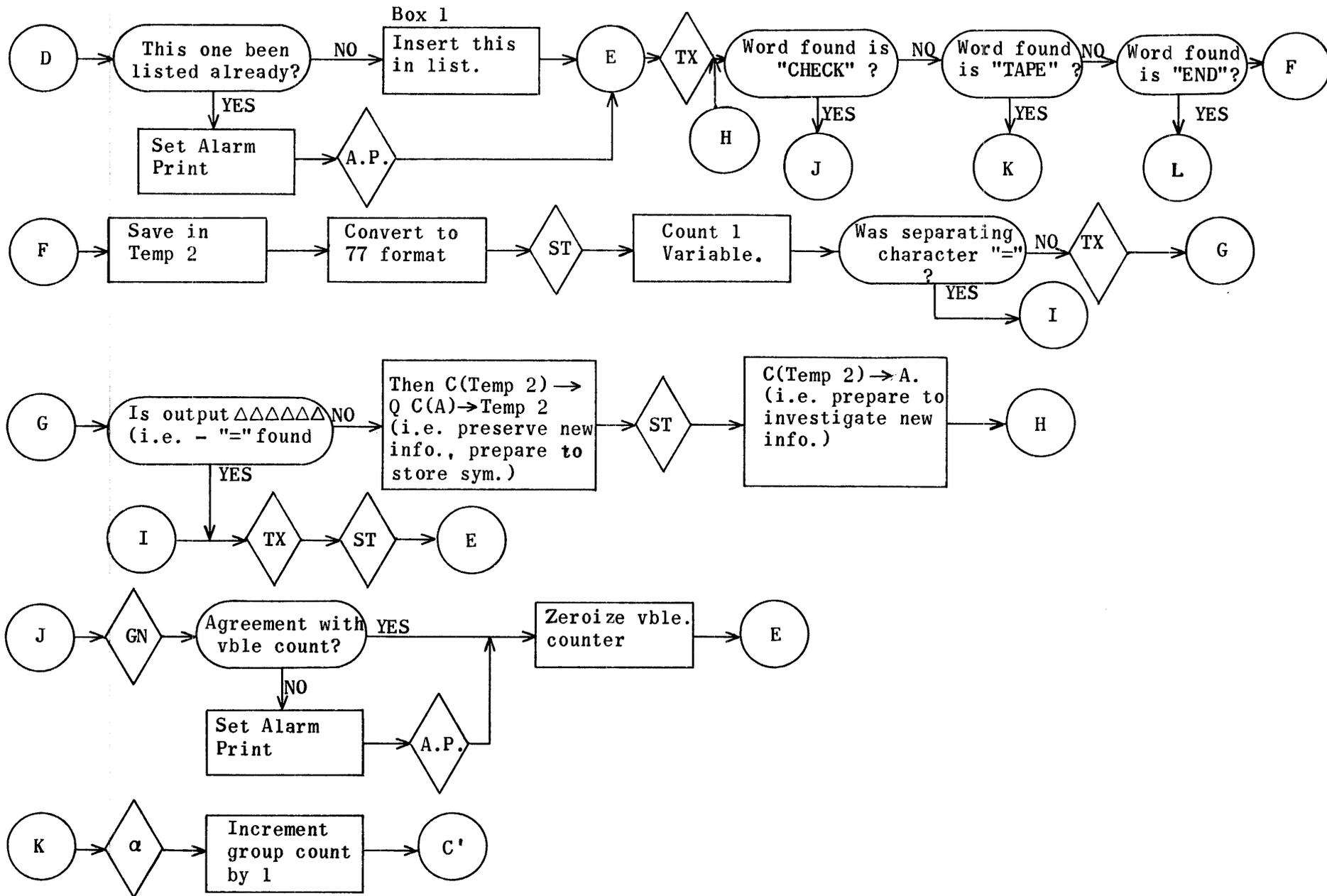
342	0 0	LN	Length of section ② of Init.	}
343	0 0	DD	Initial running address of Section ②	
344	0 ID	0	Stored here	}
345	0 0	LC	Length of Control (excluding Seg. Table)	
346	0 0	1	Initial running address	
347	0 ON	0	Stored here	}
350	0 0	LT	Length of Segment Table	
351	0 ST	0	Stored (formed) here	
352	0 CL	0	Const. Pool stored here	}
353	[0 30000 CA GI354	30000]	Indicator	

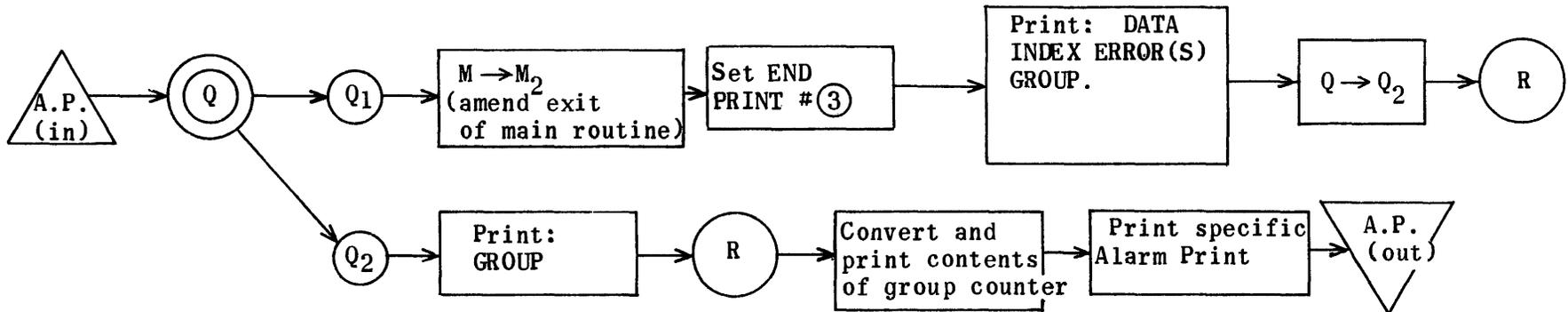
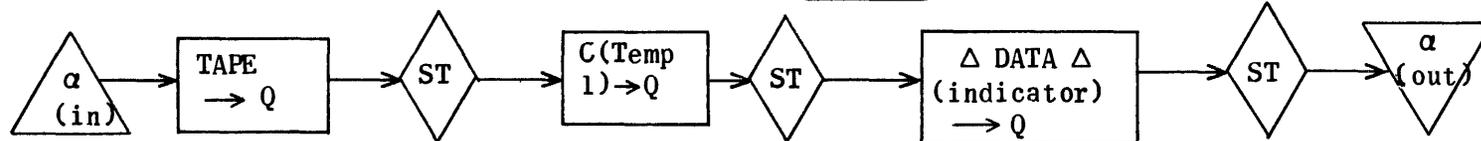
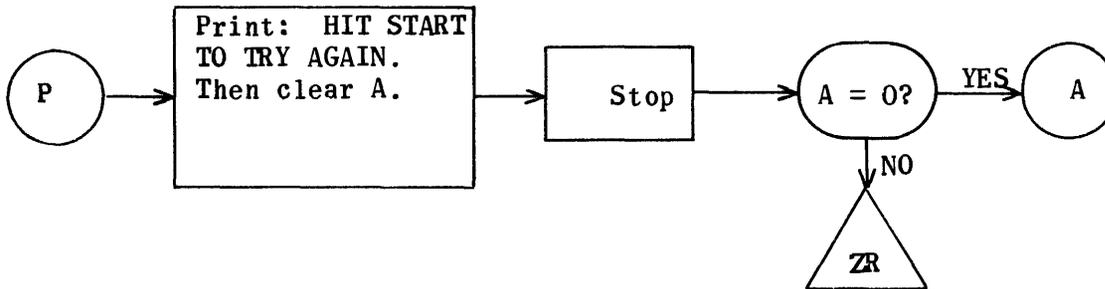
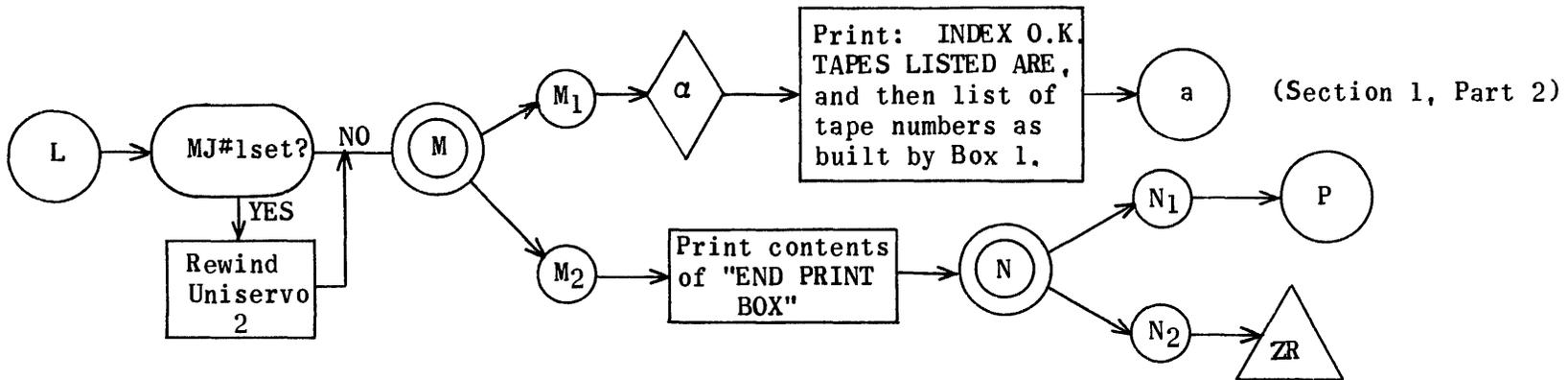


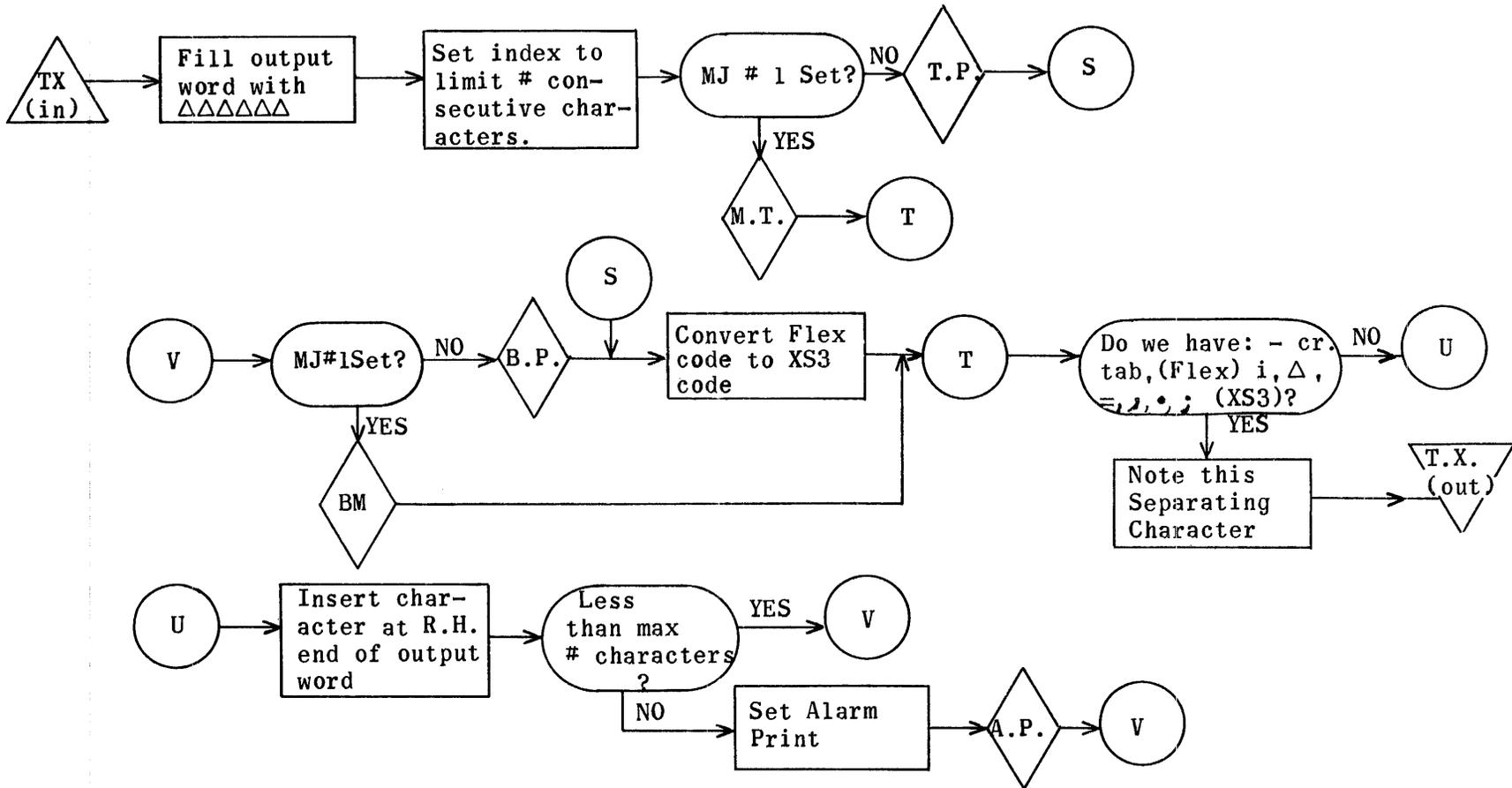
Flow Chart, General Layout of Section 1 of Initialization

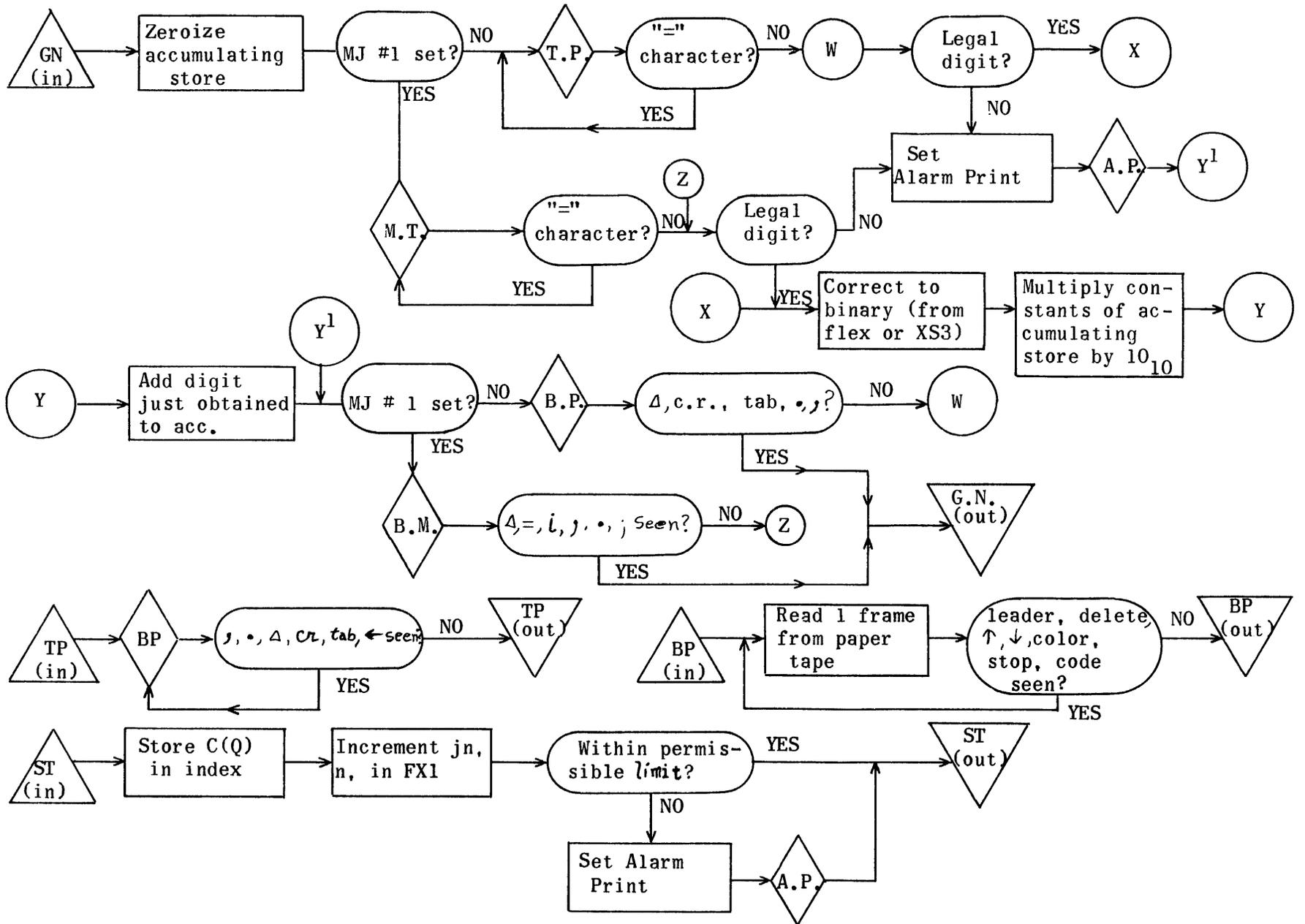


Section 1 of Initialization, Part 1

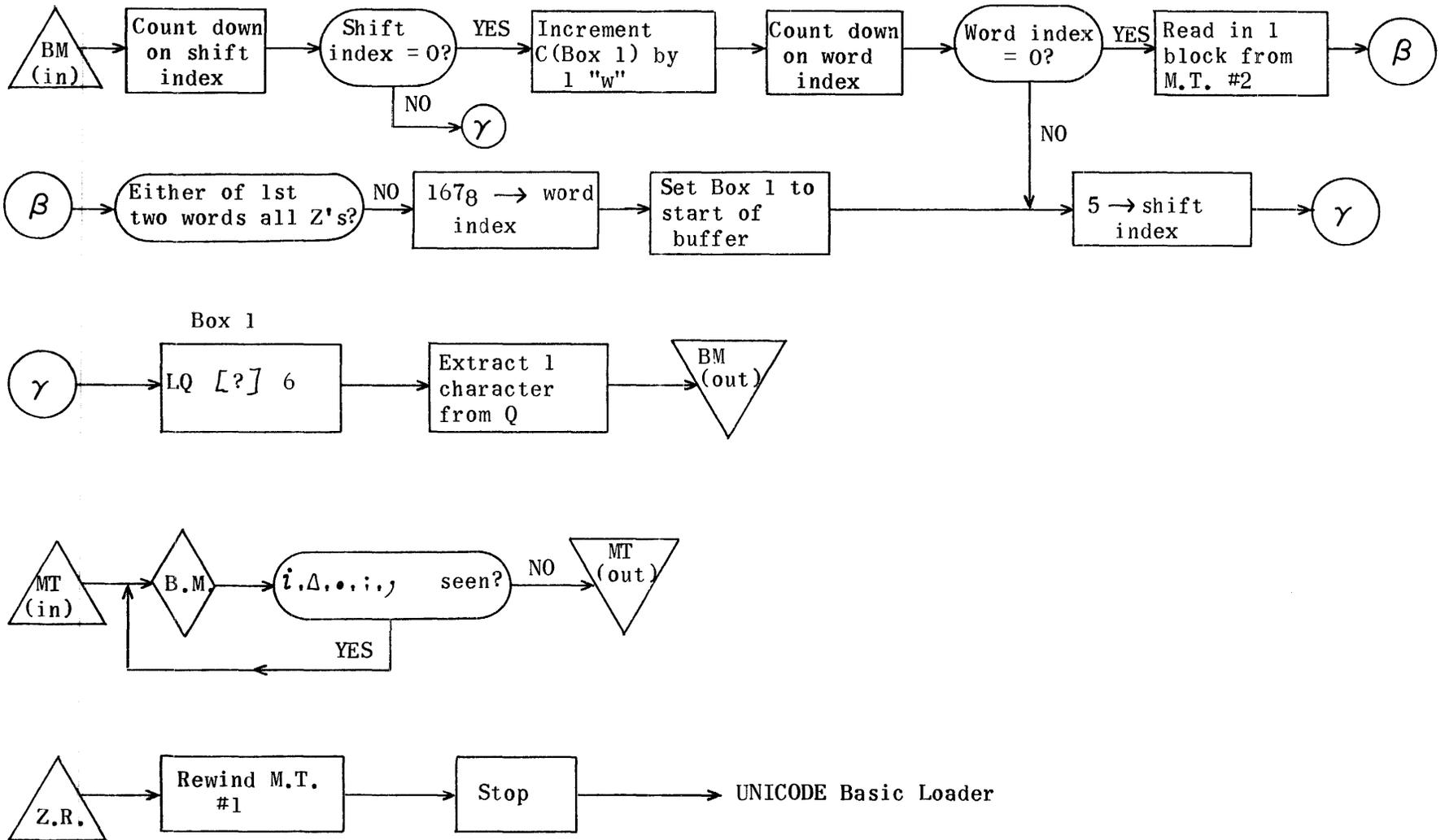






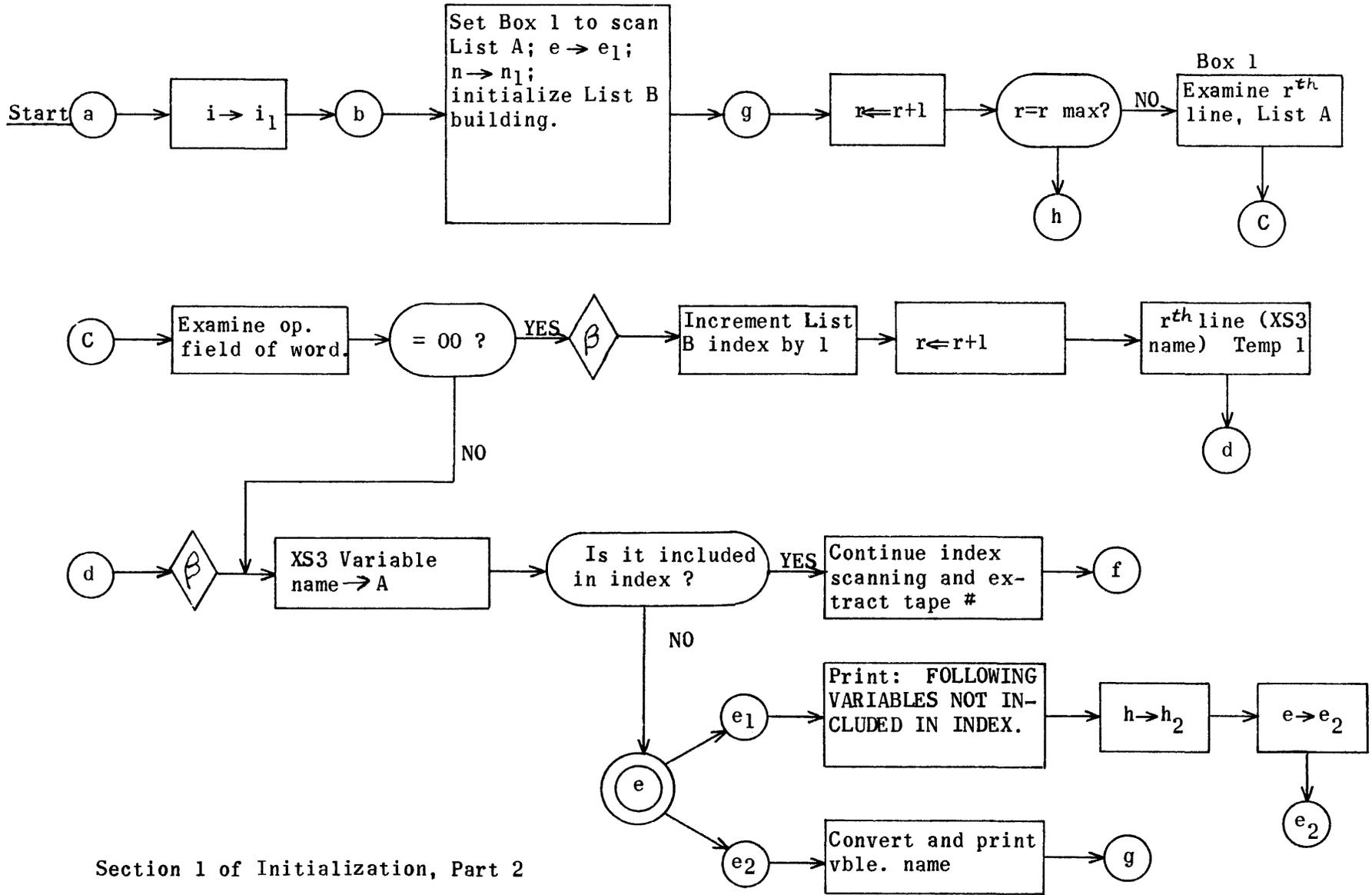


Flow Chart (cont) Section 1 of Initialization Part 1

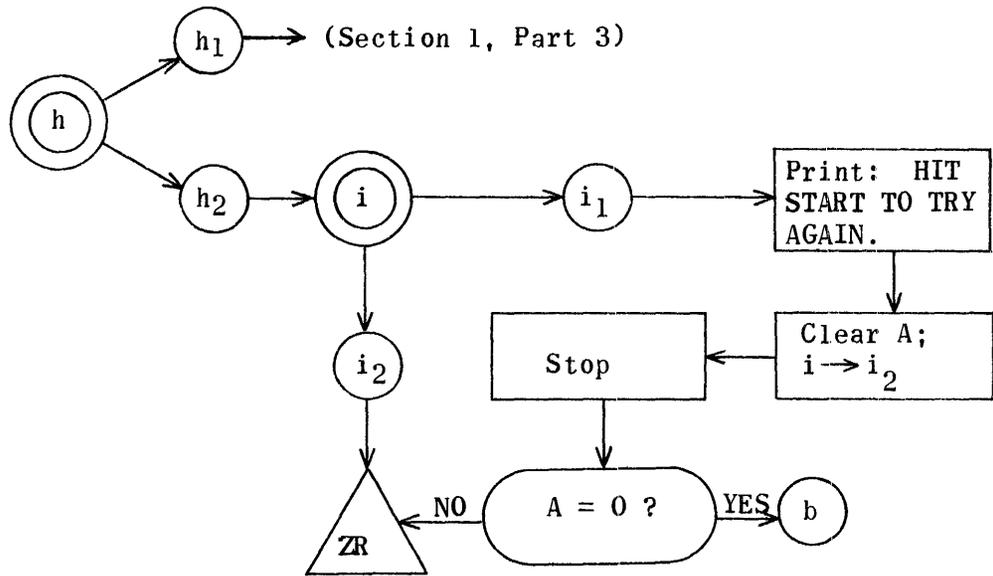
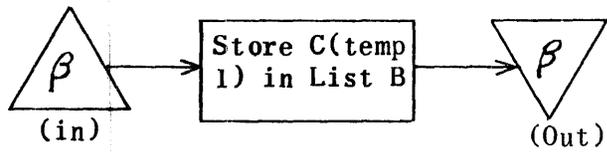
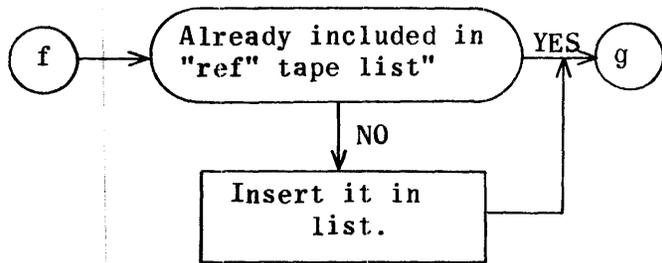


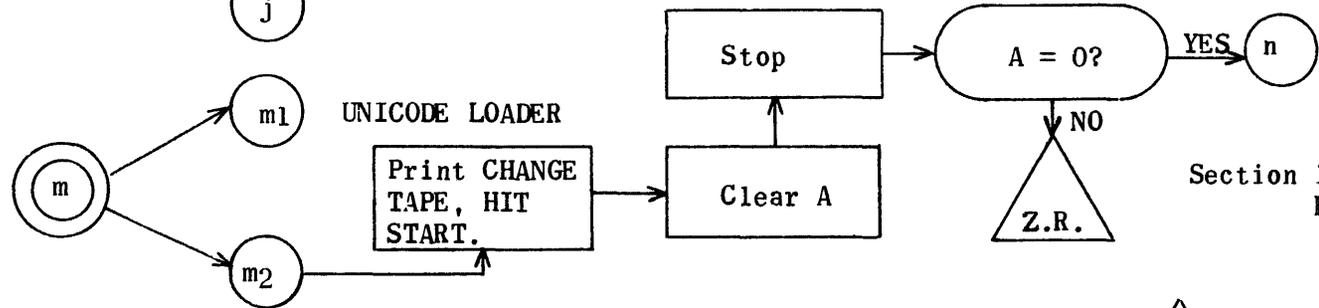
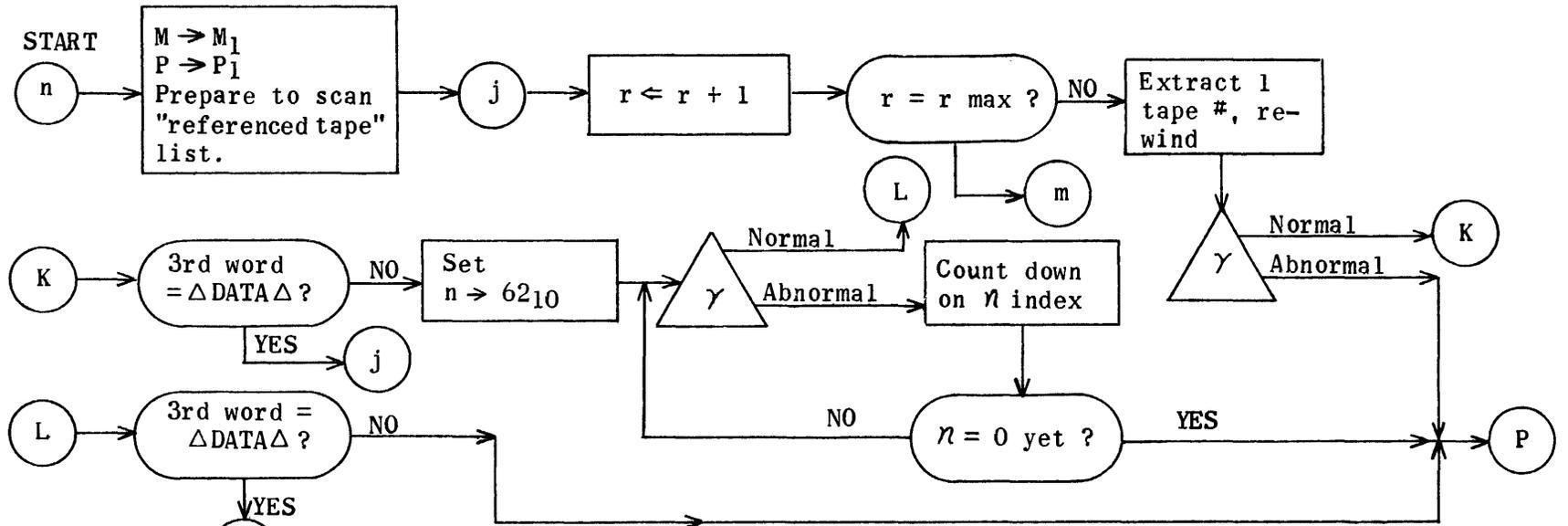
Flow Chart (cont.)

Section 1 of initialization, Part 1

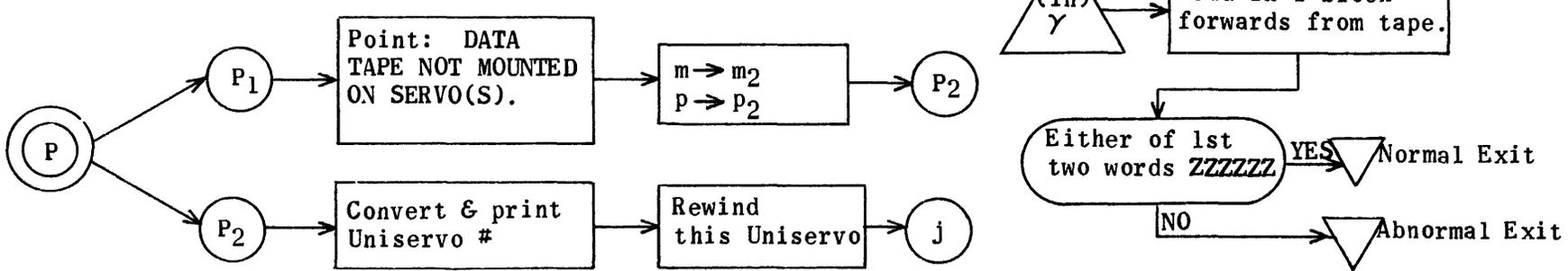


Section 1 of Initialization, Part 2





Section 1 of Initialization, Part 3



Section 1 of Initialization

	RE IA2354		Loading Address	
	RE IN2000		(27)	} Operating address of program (Total length = 1104 ₈ words)
	RE MR2027		(171)	
	RE TX2220		(37)	
	RE GN2257		(34)	
	RE BP2313		(5)	
	RE TP2320		(3)	
	RE ST2323		(14)	
	RE BM2337		(24)	
	RE MT2363		(3)	
	RE DP2366		(14)	
	RE YW2402		(12)	
	RE XW2414		(113)	
	RE YH2527		(5)	
	RE XH2534		(51)	
	RE CN2605		(102)	
	RE CL2707		(40)	
	RE LP2747		(132)	
	RE ZR3101		(3)	
	RE LA3104			List A
	RE LB7475			List B
	RE TL660		(12)	} Temporaries in Termination Buffer
	RE TM700		(5)	
	RE TB1			Buffer
	RE GT210			Tape Handler during Object Program
	RE FX1000			Fixed I/O locations.
	RE PR77250			Flex print Routine
	RE DA77300			Object Program Loader
	RE LD1500			Running address of Loader.
	IA IA			
IN	0	TV CN2	MR132	N → N ₁ (Enable 2nd pass)
	1	TV FX	ST1	} Initialize ST
	2	TP CN4	ST12	
	3	TP CL	ST13	Zeroize vble. counter
	4	TP CN10	FX1	Start FX1 with [0 20000 0]
	5	TV CN40	MR152	Q → Q ₁
	6	TV CN41	MR107	M → M ₂
	7	TP CN4	TM3	1 Group counter
	10	TP CN10	TL	Initialize Tape List Index
	11	TP LP4	MR150	Set EP ①
	12	MJ 10000	IN16	P or M?
	13	EF 0	BP4	} P - Throw away 1 frame
	14	ER 0	A	
	15	MJ 0	MR1	→ Main Routine
	16	EF 0	CN100	Mag. tape. Rewind #2
	17	TP CN77	GT3	} Read 1 block forward
	20	RJ GT2	GT	
	21	TP CN61	A	Z----Z → A

	22	EJ TB	IN25	} Check for Sentinel	
	23	EJ TB1	IN25		
	24	MJ 0	MR104		
	25	RJ BM	BM11		
	26	EJ CN4	MR1		
		CA IA27		(= MR)	
		IA IA27			
MR	0	MJ 0	BM1		
	1	TP CN54	TX2	$\infty \rightarrow$ TX index	
	2	RJ TX	TX3	Get 1 word	
	3	EJ CN63	MR7	INPUT?	
	4	EJ CN65	MR7	(VARIABLES?)	
	5	EJ CN67	MR104	END?	
	6	MJ 0	TX3		
	7	TP LP11	MR150	Set EP ②	
	10	RJ TX	TX3	Get next word.	
	11	EJ CN71	MR14	TAPE?	
	12	EJ CN67	MR104	END?	
	13	MJ 0	TX3	No.	
	14	TV CN43	MR107	TAPE seen. $M \rightarrow M_1$	
	15	TP CN70	TX2	$6 \rightarrow$ TX index	
	16	RJ GN	GN1		
	17	TJ CL4	MR21	Should not be < 2 (i.e. - <u>don't</u>	
				jump if good)	
	20	TJ CN60	MR24	Should not be ≥ 11 (i.e. - <u>do</u> jump	
				if good)	
	21	TP LP53	MR151	} Alarm (illegal tape #)	
	22	RJ MR170	MR152		
	23	MJ 0	MR37		
	24	TP A	TM	Preserve in Temp. 1	
	25	TU TL	MR26	Set up RP	
	26	RP [0]	MR33	} Search tape list to see if already	
	27	EJ TL1	MR30		referenced (if so - error)
	30	TP LP60	MR151		} Alarm (Duplicate tape #)
	31	RJ MR170	MR152		
	32	MJ 0	MR37		
	33	TV TL	MR35		
	34	RA MR35	CN101	(by L(TL1) v)	
	35	TP TM	[30000]		
	36	RA TL	CN47	by lu, lv	
	37	RJ TX	TX3	Find 1 word	
		CA IA67			
		IA IA67			
MR	40	EJ CN73	MR73	= CHECK?	
	41	EJ CN71	MR101	= TAPE?	
	42	EJ CN67	MR104	= END?	
	43	TP A	TM1	No - preserve in case needed as	
				synonym	
	44	TP A	TM2	Now count to 77 format. Send to	
				working store.	
	45	TP CN53	Q	Op field mask \rightarrow Q	

	46	QT	TM2	A	Inspect 1st two digits
	47	EJ	CN54	MR52	= 01?
	50	TP	TM2	Q	No, result to Q, all finished
	51	MJ	0	MR55	
	52	QS	CN53	TM2	Yes, replace by 77
	53	LQ	TM2	6	and shift left
	54	MJ	0	MR45	
	55	RJ	ST11	ST	Now Q holds 77 format. Store
					(counting 1 vble)
	56	SP	TX1	0	Separating symbol → A
	57	EJ	CL21	MR67	= ?
	60	RJ	TX	TX3	No, get more information
	61	EJ	CN55	MR67	Is output Δ _____ Δ ?
	62	TP	TM1	Q	No, so = was not seen. Prepare to
					store previous, meanwhile preserving
					new information
	63	TP	A	TM1	
	64	RJ	ST11	ST1	Store old name
	65	TP	TM1	A	
	66	MJ	0	MR40	Now go investigate new information
	67	RJ	TX	TX3	= seen. Obtain synonym
	70	TP	A	Q	and store it
	71	RJ	ST11	ST1	
	72	MJ	0	MR37	Then back to look for more.
	73	RJ	GN	GN1	Obtain check #
	74	EJ	ST13	MR77	Correct?
	75	TP	LP64	MR151	No. } Alarm
	76	RJ	MR170	MR152	
	77	TP	CL	ST13	Zeroize check counter
		CA	IA127		
		IA	IA127		
MR	100	MJ	0	MR37	
TAPE	101	RJ	MR147	MR141	Act appropriately
	102	RA	TM3	CN4	Up group count by 1
	103	MJ	0	MR16	and back for tape #
END	104	MJ	10000	MR106	P. or M.?
	105	MJ	0	MR107	P - jump.
	106	EF	0	CN100	M - rewind # 2
	107	MJ	0	[30000]	
	110	RJ	MR147	MR141	Act appropriately
	111	TP	LP40	PR3	
	112	RJ	PR2	PR	INDEX OK. TAPES LISTED ARE ↓
	113	TP	CL	TM2	Zeroize index
	114	TV	TL	TM2	Set it up.
	115	TU	MR27	MR120	Initialize
	116	IJ	TM2	MR120	Jump on index
	117	MJ	0	MR124	All through-out
	120	TP	[30000]	A	1 tape # to A
	121	RJ	DP13	DP	Print it
	122	RA	MR120	CN50	Increment by 1 "u"
	123	MJ	0	MR116	Back for more
	124	PR	0	CL7	Period

Z
 Z₁
 (good)

	125	PR 0	CL11	Carriage return
	126	PR 0	CL11	Carriage return
	127	MJ 0	YW	Exit (Normal)
(Z ₂)	130	TP MR150	PR3	EP Box
(bad)	131	RJ PR2	PR	
	132	RJ MR132	[30000]	(Initially MR 134)
	133	MJ 0	ZR	Exit (Get off machine)
	134	TP LP32	PR3	
	135	RJ PR2	PR	HIT START TO TRY AGAIN, ETC.
	136	SP CL	0	Clear A
	137	MS 0	MR140	Stop
		CA IA167		
		IA IA167		
MR	140	ZJ ZR	IN1	(Non zero - get off machine exit)
	141	TP CN75	Q	Δ TAPE Δ → Q
	142	RJ ST11	ST1	→ index
	143	TP TM	Q	Tape # → Q
	144	RJ ST11	ST1	→ index
	145	TP CL35	Q	
	146	RJ ST11	ST1	leave room (cleared) for indicator
	147	MJ 0	[30000]	
Error	150	[0 30000	30000]	EP parameter.
Rtne.	151	[0 30000	30000]	REP parameter.
(M)	152	MJ 0	[30000]	Entry. Initially MR 153
	153	TV CN41	MR107	M → M ₂
	154	TP LP16	MR150	Set EP (3)
	155	TP LP46	PR3	Prepare for Print-out
	156	RJ MR152	MR160	Q → Q ₂
(M ₂)	157	TP LP47	PR3	
	160	RJ PR2	PR	Go print
	161	SP TM3	0	Group count → A
	162	PR 0	CL3	Shift down
	163	RJ DP13	DP	Print group count.
	164	PR 0	CL10	Space
	165	PR 0	CL2	Shift up
	166	TP MR151	PR	Specific diagnosis
	167	RJ PR2	PR	
	170	MJ 0	[30000]	Exit
		CA IA220		
		IA IA220		
TX	0	MJ 0	[30000]	Exit.
	1	[0 30000	30000]	Output line (for cut-off symbol)
	2	[0 30000	30000]	Input line (value for index setting)
	3	TP CN55	TX36	Entry. Fill word with Δ ---- Δ
	4	TP TX2	TM2	Set index
	5	MJ 10000	TX10	P or M?
	6	RJ TP1	TP	P. Find start
	7	MJ 0	TX14	Go translate
	10	RJ MT1	MT	M. Find start
	11	MJ 0	TX21	
	12	MJ 10000	TX20	P or M?
	13	RJ BP2	BP	P - Find next frame.
	14	SA CN	17	
	15	TU A	TX16	Translate
	16	TP [30000]	A	
	17	MJ 0	TX21	

	20	RJ BM	BM1	M - Find next character
	21	RP 20006	TX23	Exit { FLEX Δ cr tab. , =
	22	EJ CL14	TX33	
	23	LQ TX36	6	XS3 i Δ = , . ;
	24	TP CL1	Q	
	25	QS A	TX36	Mask → Q
	26	IJ TM2	TX12	Insert new character
	27	TP LP70	MR151	Alarm
	30	RJ MR170	MR152	
	31	TP CN54	TM2	Reset index to large value
	32	MJ 0	TX12	
	33	TP A	TX1	
	34	TP TX36	A	Output to A
	35	MJ 0	TX	
	36	[O 30000	30000]	Word assembly space
		CA IA257		
		IA IA257		
GN	0	MJ 0	[30000]	Exit
	1	TP CL	GN33	Zeroize working store
	2	MJ 10000	GN7	P or M?
	3	RJ TP1	TP	P. Get 1st character
	4	EJ CN32	TP	Throw away =
	5	RP 20012	GN30	Check down digit list
	6	EJ CL22	GN14	
	7	RJ MT1	MT	Mag. Tape - Find 1st character.
	10	EJ CL21	MT	Throw away =
	11	TJ CN56	GN13	Should be < 15g
	12	MJ 0	GN30	
	13	ST CN37	Q	Subtract 3 → Q
	14	SP GN33	2	Multiply previous by 10
	15	SA GN33	1	
	16	QA CL33	GN33	Add in new figure
	17	MJ 10000	GN23	P or M?
	20	RJ BP2	BP	P - get next ch.
	21	RP 20005	GN5	Exit if Δ cr tab. , FLEX
	22	EJ CL6	GN26	
	23	RJ BM	BM1	M. Get next ch.
	24	RP 20006	GN11	Exit if Δ = i , . ; XS3
	25	EJ CL14	GN26	
	26	SP GN33	0	Result → A
	27	MJ 0	GN	and out.
	30	TP LP74	MR151	Alarm
	31	RJ MR170	MR152	
	32	MJ 0	GN17	
	33	[O 30000	30000]	Erasable
		CA IA313		
		IA IA313		
BP	0	EF 0	BP4	
	1	ER 0	A	

	2	RP 20006	[30000]	}	Basic paper tape read.		
	3	EJ CL	BP				
	4	IO 3	0				
		CA IA320					
		IA IA320					
TP	0	RJ BP2	BP	}	Throw away routine. (After this, we have a significant code)		
	1	RP 20006	[30000]				
	2	EJ CL6	BP				
		CA IA323					
		IA IA323					
ST	0	RA ST13	CN4	}	Count 1 vble. Build index Inc. st. order Alarm. Increment Vble. count.		
	1	TP Q	[30000]				
	2	RA ST1	ST12				
	3	RA FX1	CN47				
	4	TJ CN76	ST11				
	5	TP LP77	MR151				
	6	RJ MR170	MR152				
	7	TP CL	ST12				
	10	TP CN10	FX1				
	11	MJ 0	[30000]				
	12	0 [30000	30000]				
	13	0 [30000	30000]				
		CA IA337					
		IA IA337					
BM	0	MJ 0	[30000]	}	Exit Entry. Jump on shift index Jump on word index Read 1 block forward Z-----Z → A If Sentinel seen., alarm. Word index → 167 Shift index → 5 Extract 1 character And out Alarm Shift index Word index		
	1	IJ BM22	BM14				
	2	RA BM14	CN50				
	3	IJ BM23	BM13				
	4	TP CN77	GT3				
	5	RJ GT2	GT				
	6	TP CN61	A				
	7	EJ TB	BM17				
	10	EJ TB1	BM17				
	11	TP CN57	BM23				
	12	TU BM7	BM14				
	13	TP CN74	BM22				
	14	LQ [30000].	6				
	15	QT CL1	A				
	16	MJ 0	BM				
	17	TP LP24	MR151				
	20	RJ MR170	MR152				
	21	MJ 0	MR104				
	22	[0 30000	30000]				
	23	[0 30000	30000]				
		CA IA363					
		IA IA363					
MT	0	RJ BM	BM1			}	Mag. Tape throw away.

	1	RP 20005	[30000]	}	(Discard i Δ , . . ;)
	2	EJ CL14 CA IA366	BM1		
DP	0	IA IA366 DV CN72	Q		Quantity given in A. Divide by 10
	1	TP A	TM4		Save remainder
	2	TN Q	A	}	Tens figure zero?
	3	ZJ DP4	DP6		
	4	AT CL34	DP5		No, form print order
	5	[0 30000	30000]		
	6	TP CL34	A		Dummy print again
	7	ST TM4	DP10		form print order for units
	10	[0 30000	30000]		
	11	PR 0	CL10	}	Then 2 spaces
	12	PR 0	CL10		
	13	MJ 0	[30000]		Exit
		CA IA402			

Initialization for XW

YW	0	IA IA402 TV XW106	XW74	}	Enable restart after 1st error pass
	1	TU FX	XW16		
	2	TU FX1	XW15	}	Set up index-scanning
	3	TV XW107	XW53		
	4	TP CN10	TL		Initialize error print-out section
	5	TV XW110	XW1		Set Tape List index to 0 20000 0
	6	TU XW106	XW2	}	Set normal exit (YH)
	7	TP LA	TM1		Scan List A from LA1
	10	TV XW111	XW50		Set index
	11	TP CL	LB		Build List B from LBl
		CA IA414			and set index

First Run-Through Data List, and Preliminary Checking.

XW	0	IA IA414		
	1	IJ TM1	XW2	Jump on List A index
	2	MJ 0	[30000]	Exit when all completed
	3	TP[30000]	A	Examine one item.
	4	TP A	TM	Save it in temp
	5	TP CN53	Q	Op field mask → Q
	6	QT A	A	and examine Op. field
	7	ZJ XW14	XW7	
		RJ XW52	XW47	Zero, . . a "mod. Ed.a" line -
	10	RA XW2	CN50	build List B
	11	TU A	XW12	So
	12	TP[30000]	TM	extract
	13	RJ XW52	XW50	next line (XS3 name)
	14	TP TM	A	and store it as well, in List B
	15	RP[0]	XW53	Now, name to A
	16	EJ[30000]	XW17	Scan index. (Alarm, if not present)
	17	SN Q	17	
	20	SA XW15	0	
	21	SA XW16	0	
	22	TU A	XW27	Set up EJ for continued search
	23	LQ Q	17	
	24	TU Q	XW26	Set up RP
	25	TP CN75	A	[Δ TAPE Δ to A
	26	RP[0]	XW103	Continue to scan index, searching for
	27	EJ[30000]	XW30	tape #
	30	SN Q	17	
	31	SA XW26	0	
	32	SA XW27	0	
	33	TU A	XW34	
	34	SP[30000]	0	Tape # → A
	35	TP A	TM	and save it
	36	TU TL	XW37	
	37	RP[0]	XW41	
		CA IA454		
XW	40	IA IA454		
	41	EJ TL1	XW45	Scan referenced tape list
	42	TV TL	XW43	Not yet present - so insert it
	43	RA XW43	CN101	
	44	TP TM	[30000]	
	45	RA TL	CN47	Increment index
	46	RA XW2	CN50	Prepare to scan further down list
	47	MJ 0	XW	
	50	RA LB	CN4	
	51	TP TM	[30000]	
	52	RA XW50	CN4	
		MJ 0	[30000]	

53	MJ 0	[30000]	Initially XW54. Error Routine.
54	TP LP110	PR3	Print: FOLLOWING VARIABLES NOT
55	RJ PR2	PR	INCLUDED IN INDEX.
56	TV XW112	XW1	Amend exit from main routine
			(to XW 73)
57	RJ XW53	XW60	
60	TP CN74	TM2	2nd and subsequent errors here.
			5 → index
61	LQ TM	6	Shift one character over
62	QT CL1	A	Extract it
63	RP 20074	XW71	
64	EJ CN1	XW65	
65	SN Q	17	Compute and print Flex-code
66	SA XW63	71	
67	PR 0	A	
70	IJ TM2	XW61	
71	PR 0	CL11	Carriage return, when fully printed
72	MJ 0	XW45	
73	PR 0	CL11	Extra CR
74	RJ XW74	[30000]	Error END. Initially XW76
75	MJ 0	ZR	2nd time - get off machine
76	TP LP32	PR3	1st time: HIT START TO TRY AGAIN
77	RJ PR2	PR	
	CA IA514		

		IA IA514	
XW	100	SP CL	0
	101	MS 0	XW102
	102	ZJ ZR	YW1
			Clear A
			Stop
			If A ≠ 0, get off machine; otherwise,
			try again
	103	TP LP131	PR3
	104	RJ PR32	PR
	105	MJ 0	ZR
	106	0 LA1	XW76
	107	0 0	XW54
	110	0 0	YH
	111	0 0	LB1
	112	0 0	XW73
		CA IA527	
			Print: MACHINE ERROR
			G-O-M
			Constants.

Initialization for XH

		IA IA527	
YH	0	TP CL	TM3
	1	TV TL	TM3
	2	TV XH46	XH1
	3	TU XW40	XH2
	4	TV XH47	XH17
		CA IA534	
			Set index
			Set normal exit: BACK TO LOADER
			Start list at TLI
			Initialize error procedures

XH	0	IA IA534		
	1	IJ TM3	XH2	Count down on index
	2	MJ 0	[30000]	Exit
	3	TP [30000]	TM	Extract 1 tape #
	4	SP TM	14	
	5	AT CL36	TM1	Rewind this tape.
	6	EF 0	A	
	7	RJ XH40	XH31	Go read in 1 block forward
	10	MJ 0	XH17	Alarm return - no lead Sentinels
	11	TP CL35	A	Δ DATA → A
	12	EJ TB2	XH27	Tape so labelled?
	13	TP CL21	TM2	No, prepare to read more. Set scan index
	14	RJ XH40	XH31	
	15	IJ TM2	XH31	No Sentinel seen
	16	TP CL35	A	Δ DATA Δ → A
	17	EJ TB2	XH27	Tape so labelled?
	20	MJ 0	[30000]	No, alarm. (initially XH20)
	21	TP LP117	PR3	Print: DATA TAPE NOT MOUNTED
	22	RJ PR2	PR	ON SERVOS Δ
	23	TV XH50	XH1	Amend exit from main routine. (to XH41)
	24	RJ XH17	XH24	Disconnect this path
	25	SP TM	0	Tape # → A
	26	RJ DP13	DP	Convert and print it
	27	EF 0	TM1	Rewind tape
	30	RA XH2	CN50	Pick next one
	31	MJ 0	XH	
	32	SP TM	17	
	33	AT CL37	GT3	Read 1 block forward
	34	RJ GT2	GT	
	35	TP CN61	A	Z-----Z to A
	36	RP 20002	XH40	If ZZ___Z seen, go to exit +1.
	37	EJ TB	XH37	If not, normal exit.
		RA XH40	CN4	
		CA IA574		

XH	40	IA IA574		
	41	MJ 0	[30000]	Error exit.
	42	TP LP125	PR3	Print: CHANGE TAPE, HIT START.
	43	RJ PR2	PR	Clear A
	44	SP CL	0	Stop
	45	MS 0	XH45	Hit Start to continue
	46	ZJ ZR	YH	
	47	0 0	LD1	
	48	0 0	XH20	
	49	0 0	XH41	
	50	CA IA605		

XS3 - Stored by Flex.

CN	0	0 0	CN	
	1	0 0	66	T
	2	0 0	MR134	
	3	0 0	51	\bar{O}
	4	0 0	01	$\bar{\Delta}$
	5	0 0	33	H
	6	0 0	50	N
	7	0 0	47	M
	10	0 20000	0	
	11	0 0	46	L
	12	0 0	54	R
	13	0 0	32	G
	14	0 0	34	I
	15	0 0	52	P
	16	0 0	26	C
	17	0 0	70	V
	20	0 0	30	E
	21	0 0	74	Z
	22	0 0	27	D
	23	0 0	25	B
	24	0 0	65	S
	25	0 0	73	Y
	26	0 0	31	F
	27	0 0	72	X
	30	0 0	24	A
	31	0 0	71	W
	32	0 0	44	J
	33	0 0	14	9
	34	0 0	67	U
	35	0 0	53	Q
	36	0 0	45	K
	37	0 0	03	O
		CA IA645		
		IA IA645		
CN	40	0 0	MR153	
	41	0 0	MR130	
	42	0 0	22	.
	43	0 0	MR110	
	44	0 0	76	=
	45	0 0	01	cr ==> Δ
	46	0 0	21	,
	47	0 1	1	
	50	0 1	0	
	51	0 0	01	tab ==> Δ
	52	0 0	04	l
	53	77 0	0	
	54	01 0	0	

55	01 01010	10101	
56	0 0	15	
57	0 0	167	
60	0 0	13	8
61	74 74747	47474	Z-----Z
62	0 0	10	5
63	01 34505	26766	Δ INPUT
64	0 0	07	4
65	34 24254	63065	IABLES
66	0 0	11	6
67	01 01013	05027	Δ Δ Δ END
70	0 0	06	
71	01 01662	45230	Δ Δ TAPE
72	0 0	12	7
73	01 26333	02645	Δ CHECK
74	0 0	05	2
75	01 66245	23001	Δ TAPE Δ
76	0 20150	150	Limit for index
77	50 102	TB	GTH code for read 1 blk. forward(#2)
100	02 00200	20000	Rewind Uniservo 2
101	0 0	TL1	Constant
	CA IA707		

CL		IA IA707					
0	0	0	0	0	Leader		
1	0	0	77	}	Delete		
2	0	0	47		↑ → basic throw aways		
3	0	0	57				
4	0	0	02		Color		
5	0	0	43		Stop		
6	0	0	46	}	Cut off's		
7	0	0	42				
10	0	0	04			Δ	
11	0	0	45			Cr	
12	0	0	51			Tab	
13	0	0	61	Backspace			
14	0	0	01	Δ	}	Flex-codes	
15	0	0	21	.			
16	0	0	22	.			XS3 codes.
17	0	0	23	.			
20	0	0	0	i			
21	0	0	76	=			
22	0	0	33	9			
23	0	0	60	8			
24	0	0	72	7			
25	0	0	66	6			
26	0	0	62	5			
27	0	0	64	4			
30	0	0	70	3			
31	0	0	74	2			
32	0	0	52	1			
33	0	0	37	0			
34	PR	0	CL33		Dummy print		
35	01	27246	62401		Δ DATA Δ		
36	02	00200	0		General rewind code		
37	50	00100	TB		General read code		
	CA	IA747					

LP		IA IA747			
0	45	47140	62220	Cr ↑	} EP ①
1	27	04140	61603	X Δ	
2	07	15122	00520	M P R E H E	
3	06	24142	31120	N S I B L E	
4	0	LP	4		} EP ②
5	45	47060	30422	Cr ↑	
6	30	01300	40130	A T A Δ T A	
7	15	20042	41520	P E Δ S P E	
10	16	14261	42022	C I F I E D	} EP ③
11	0	LP5	4		
12	45	01052	01220	Cr T H E R E	
13	26	03122	00434	F O R E Δ U	
14	06	30161	62015	N A C C E P	}
15	01	30231	12000	T A B L E -	
16	0	LP12	4		
17	45	47200	62204	Cr ↑	E N D Δ

20	06	03010	43112	N	O	T	△	W	R	}	BM Special alarm
21	14	01012	00604	I	T	T	E	N	△		
22	03	06040	13015	O	N	△	T	A	P		
23	20	45450	00000	E	Cr	Cr					
24	0	LP17	5								
25	04	04040	40514	△	△	△	△	H	I		
26	01	04240	13012	T	△	S	T	A	R		
27	01	04010	30401	T	△	T	O	△	T		
30	12	25043	01330	R	Y	△	A	G	A		
31	14	06454	50000	I	N	Cr	Cr	—			
32	0	LP25	5								
33	45	47140	62220	Cr	↑	I	N	D	E		
34	27	04033	60404	X	△	O	K	△	△		
35	01	30152	02404	T	A	P	E	S	△		
36	11	14240	12022	L	I	S	T	E	D		
37	04	30122	00457	△	A	R	E	△	↓		
	CA	IA1007									

IA IA1007

LP

40	0	LP33	5
41	45	47223	00130
42	04	14062	22027
43	04	20121	20312
44	24	45454	50000
45	13	12033	41504
46	0	LP41	5
47	0	LP45	1
50	14	11112	01330
51	11	04013	01520
52	04	06034	54500
53	0	LP50	3
54	22	34151	11416
55	30	01200	40130
56	15	20040	60345
57	45	0	0
60	0	LP54	4
61	16	05201	63604
62	14	06160	31212
63	20	16014	54500
64	0	LP61	3
65	31	03122	20401
66	03	03041	10306
67	13	45450	0
70	0	LP65	3
71	14	11112	01330
72	11	04221	41314
73	01	45450	0
74	0	LP71	3
75	03	17201	22611
76	03	31454	50000
77	0	LP75	2
CA		IA1047	

Cr	↑	D	A	T	A	
△		I	N	D	E	X
△		E	R	R	O	R
S	Cr	Cr	Cr	-	-	
G	R	O	U	P	△	

Error
print-out
headings

I	L	L	E	G	A
L	△	T	A	P	E
△	N	O	Cr	Cr	-
D	U	P	L	I	C
A	T	E	△	T	A
P	E	△	N	O	Cr
Cr					

①

②

C	H	E	C	K	△
I	N	C	O	R	R
E	C	T	Cr	Cr	-

③

W	O	R	D	△	T
O	O	△	L	O	N
Cr	Cr	Cr	-		

④

I	L	L	E	G	A
L	△	D	I	G	I
T	Cr	Cr			

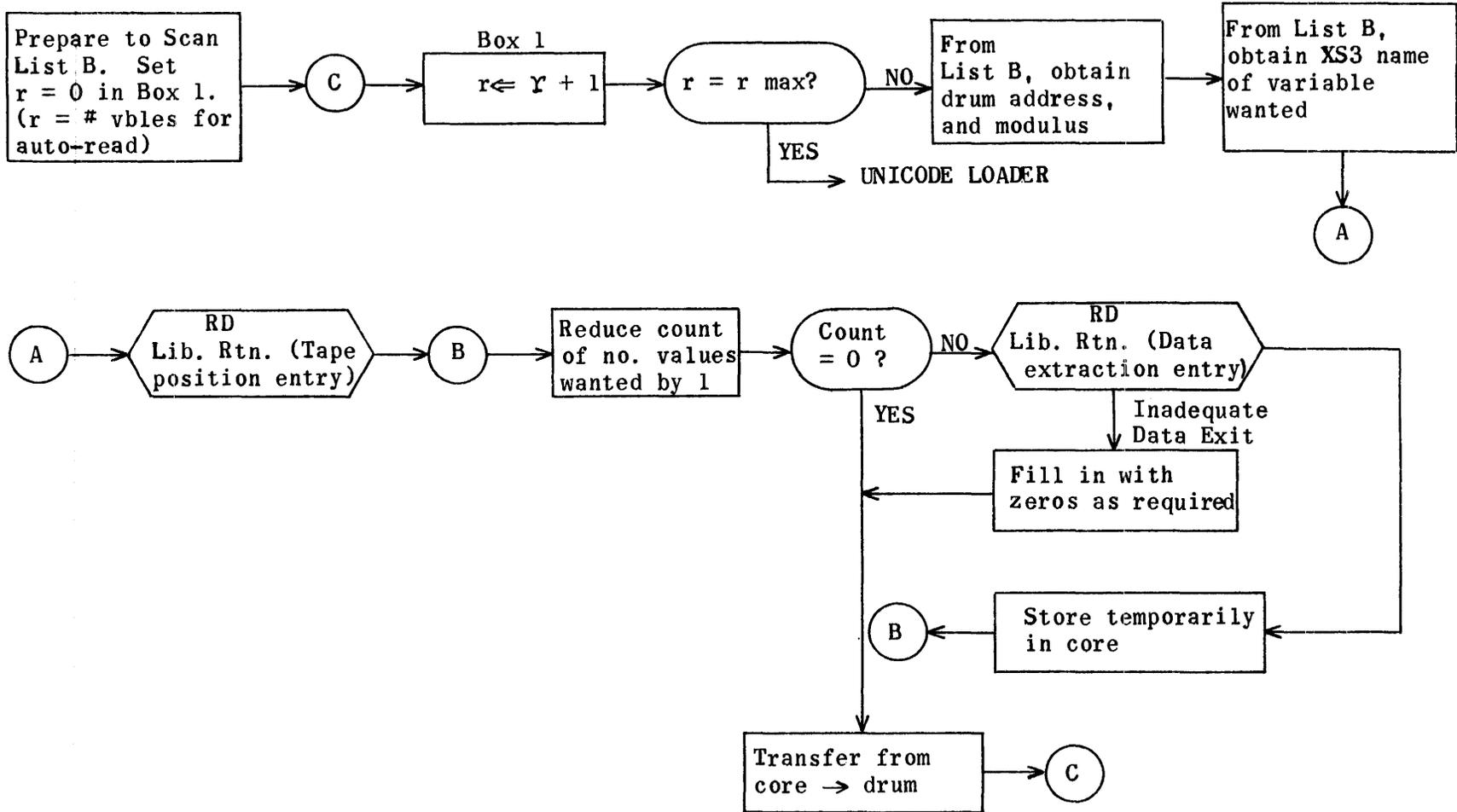
O	V	E	R	F	L
O	W	Cr	Cr	-	

		IA	IA1047			
LP	100	45	47260	31111	Cr ↑	F O L L
	101	03	31140	61304	O W	I N G Δ
	102	17	30121	43023	V A	R I A B
	103	11	20240	40603	L E	S Δ N O
	104	01	04140	61611	T Δ	I N C L
	105	34	22202	20414	U D	E D Δ I
	106	06	04140	62220	N Δ	I N D E
	107	27	45450	0	X Cr Cr	
	110	0	LP100	10		
	111	45	47223	00130	Cr ↑	D A T A
	112	04	01301	52004	Δ T	A P E Δ
	113	06	03010	40703	N O	T Δ M O
	114	34	06012	02204	U N	T E D Δ
	115	03	06042	42012	O N	Δ S E R
	116	17	03240	40457	V O	S Δ Δ ↓
	117	0	LP111	6		
	120	45	45471	60530	Cr Cr ↑	C H A
	121	06	13200	40130	N G	E Δ T A
	122	15	20040	40405	P E	Δ Δ Δ H
	123	14	01042	40130	I T	Δ S T A
	124	12	01454	50000	R T	Cr Cr
	125	0	LP120	5		
	126	45	47073	01605	Cr ↑	M A C H
	127	14	06200	42012	I N	E Δ E R
	130	12	03124	54500	R O	R Cr Cr -
	131	0	LP126	3		
		CA	IA1101			

Box 2
errors

		IA	IA1101		
ZR	0	EF	0	ZR2	Rewind tape # 1
	1	MS	0	DA	Back to basic loader
	2	02	00200	10000	
		CA	IA1104		

Section 2 of Initialization



Regional Definitions for Section 2 of Initialization.

RE	ID3460	Loading address	
RE	DD1750	(46)	} Operating addresses of program (total length = 614 ₈ words)
RE	IN2016	(12)	
RE	DR2030	(67)	
RE	BM2117	(30)	
RE	ST2147	(15)	
RE	PS2164	(65)	
RE	SC2251	(10)	
RE	MF2261	(7)	
RE	TB2270	(12)	
RE	EP2302	(17)	
RE	GG2321	(170)	
RE	CF2511	(53)	
RE	BF1	Buffer	
RE	GT210	Tape Handler during Object Program	
RE	FX1000	Fixed I/O locations	
RE	TN660	(1)	} Temporaries in Termination Buffer
RE	XX661	(5)	
RE	CC666	(27)	
RE	IL2571	2500 ₁₀ words of intermediate storage	
RE	LB7475	List B	
RE	PR77250	Flex print routine	
RE	LD1500	Operating address of Object Program Loader	

Automatic Data Read-in— Section 2 of Initialization

DD	0	IA ID		
	0	TP LB	DD44	Set up List B index
	1	TU DD35	DD5	} Initialize reading of List B
	2	TU DD36	DD13	
	3	IJ DD44	DD5	
	4	MJ 0	LD1	Count down on List index
	5	TP [30000]	Q	All through - Exit - <u>BACK TO LOADER</u>
	6	TV Q	DD31	1st of line pair (mod, da) → Q
	7	QT DD37	DD45	Set drum address
	10	AT DD40	DD30	Extract modulus
	11	LQ DD45	25	and form drum loading RP
	12	TV DD35	DD21	Shift to "v" to form index
	13	TP [30000]	IN1	Initialize
	14	RJ IN	IN2	Name of variable wanted?
	15	MJ 0	DD13	Position tape
	16	IJ DD45	DD20	EOD exit - should never come up
	17	MJ 0	DD30	OK - no count down on quantity req'd
	20	RJ IN	IN3	Obtain 1 word
	21	TP Q	[30000]	Store temporarily in core
	22	RA DD21	DD42	
	23	MJ 0	DD16	
	24	TV DD21	DD25	Here on inadequate data - fill with zero
	25	TP DD41	[30000]	
	26	RA DD25	DD42	
	27	IJ DD45	DD25	
	30	[0 30000	30000]	} OK - transfer to drum
	31	TP IL	[30000]	
	32	RA DD5	DD43	
	33	RA DD13	DD43	
	34	MJ 0	DD3	Back for more
	35	0 LB1	IL	
	36	0 LB2	0	
	37	0 07777	0	
		CA ID40		

		IA	ID40		
DD	40	RP	30000	DD32	
	41	0	0	0	
	42	0	0	1	
	43	0	2	0	
	44	[0	30000	30000]	Erasable. List B index
	45	[0	30000	30000]	Erasable. Quantity index
		CA	ID46		

The Read Permanent Library Subroutine is inserted from ID46 through ID350. Annotated coding for this subroutine can be found in Section II, 2, b, of this manual.

From ID351 on, the Excess-Three Decimal to Floating Point routine is inserted. This routine is flow charted and explained in Section III, 3, a, under Translation Subroutines.

Control Section for Object Program

During the execution of the Object Program the Control section is entered through F₂ as a result of an Interpret (IP) command. The IP command is used in the Object Program to provide the required information for suitable transfer of control from one segment to another. The form of the IP command is:

```
14 OFFTT XXXXX
```

Where FF is the number of the segment containing the IP command,

TT is the number of the segment to which control is to be transferred, and

XXXXX is the address in segment TT receiving control.

Although there is no actual segment numbered 0, an IP command with FF = 0 and TT = 1 is built by Initialization Generation to provide the starting point for Segment 1. Thus, when MS₂ is set, which provides a computer stop at the end of a segment, a stop will also occur at the end of the imaginary Segment 0 and preceding the read-in of Segment 1. There is, of course, no Termination coding for the imaginary Segment 0.

When control is entered it performs the following tasks:

- 1) Reads in and executes Termination (if any) for Segment FF.
- 2) Moves Object Program tape to Segment TT.
- 3) Reads Segment TT and its Preface (if any) to H.S.S.
- 4) Executes Preface (if any) for Segment TT.
- 5) Transfers control into Segment TT at XXXXX given in IP command.

The Move Tape subroutine is dependent on the Segment Table, built by the Segmentation Phase, to determine the correct block count in moving the Object Program tape from Segment FF to Segment TT. The Segment Table is always 17_{10} words in length, as follows:

TBO	0 0 0	0 0 0	0 X X	X X X
1	B (0)	B(16)	B(32)	B(48)
2	B (1)	B(17)	B(33)	B(49)
3	B (2)	B(18)	B(34)	B(50)
4	B (3)	B(19)	B(35)	B(51)
5	B (4)	B(20)	B(36)	B(52)
6	B (5)	B(21)	B(37)	B(53)
7	B (6)	B(22)	B(38)	B(54)
10	B (7)	B(23)	B(39)	B(55)
11	B (8)	B(24)	B(40)	B(56)
12	B (9)	B(25)	B(41)	B(57)
13	B(10)	B(26)	B(42)	B(58)
14	B(11)	B(27)	B(43)	B(59)
15	B(12)	B(28)	B(44)	B(60)
16	B(13)	B(29)	B(45)	B(61)
17	B(14)	B(30)	B(46)	B(62)
20	B(15)	B(31)	B(47)	B(63)

XXXXX = address to which all segments of the problem are read from tape.

B(K) denotes the total number of blocks on the Object Program tape required by Segment K. This includes the Label block, the Segment, the Preface, and the Termination blocks.

$B(0) = 0$

$B(K) = 0$ if $K >$ the total number of segments in the problem.

The following conditions are assumed for FF and TT

$$\begin{aligned} 0 &\leq FF \leq 63_{10} \\ 1 &\leq TT \leq 63_{10} \\ FF &\neq TT \end{aligned}$$

To move the tape from Segment FF to Segment TT, two cases must be considered.

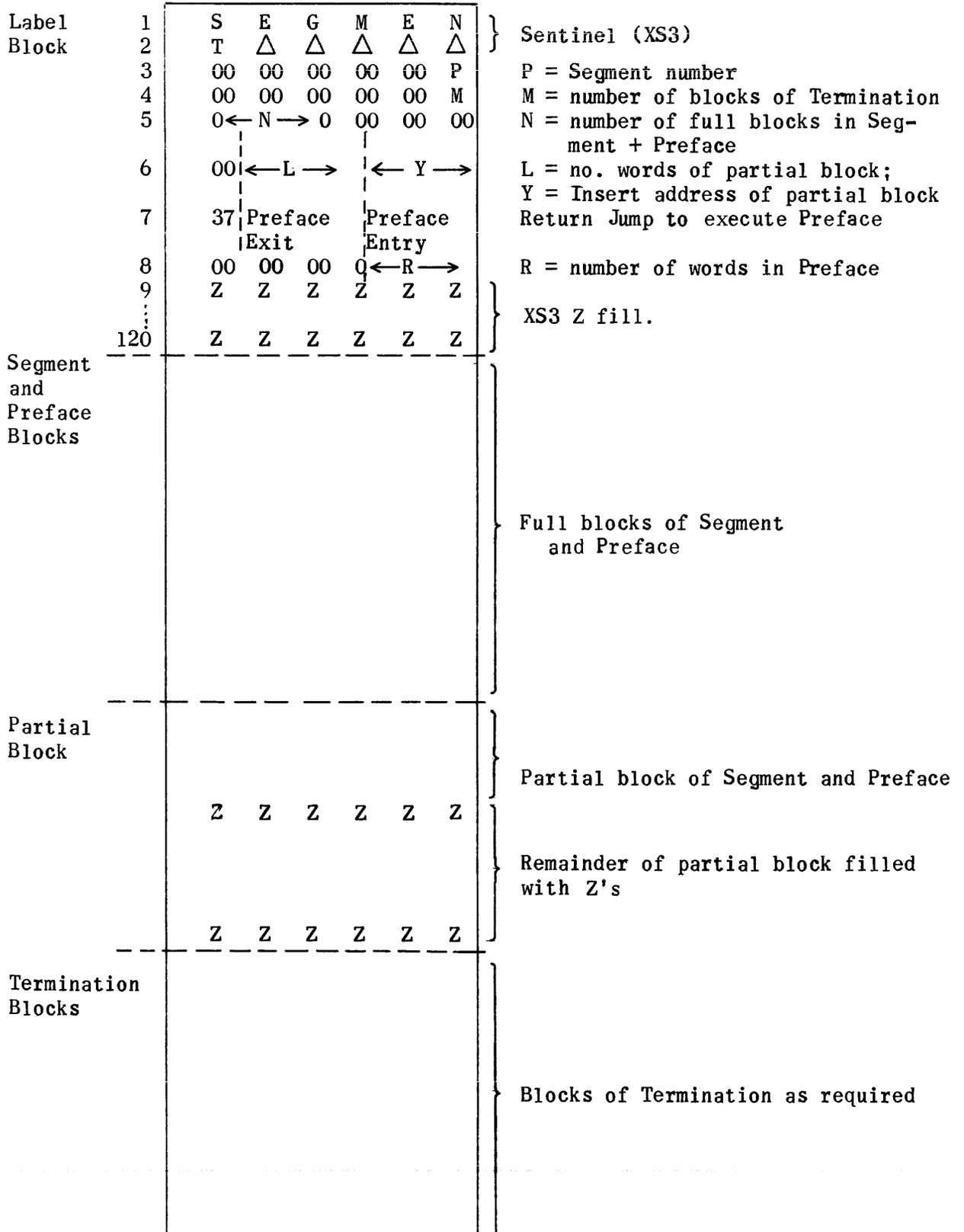
- Case 1 : $FF < TT$
- Case 2 : $FF > TT$

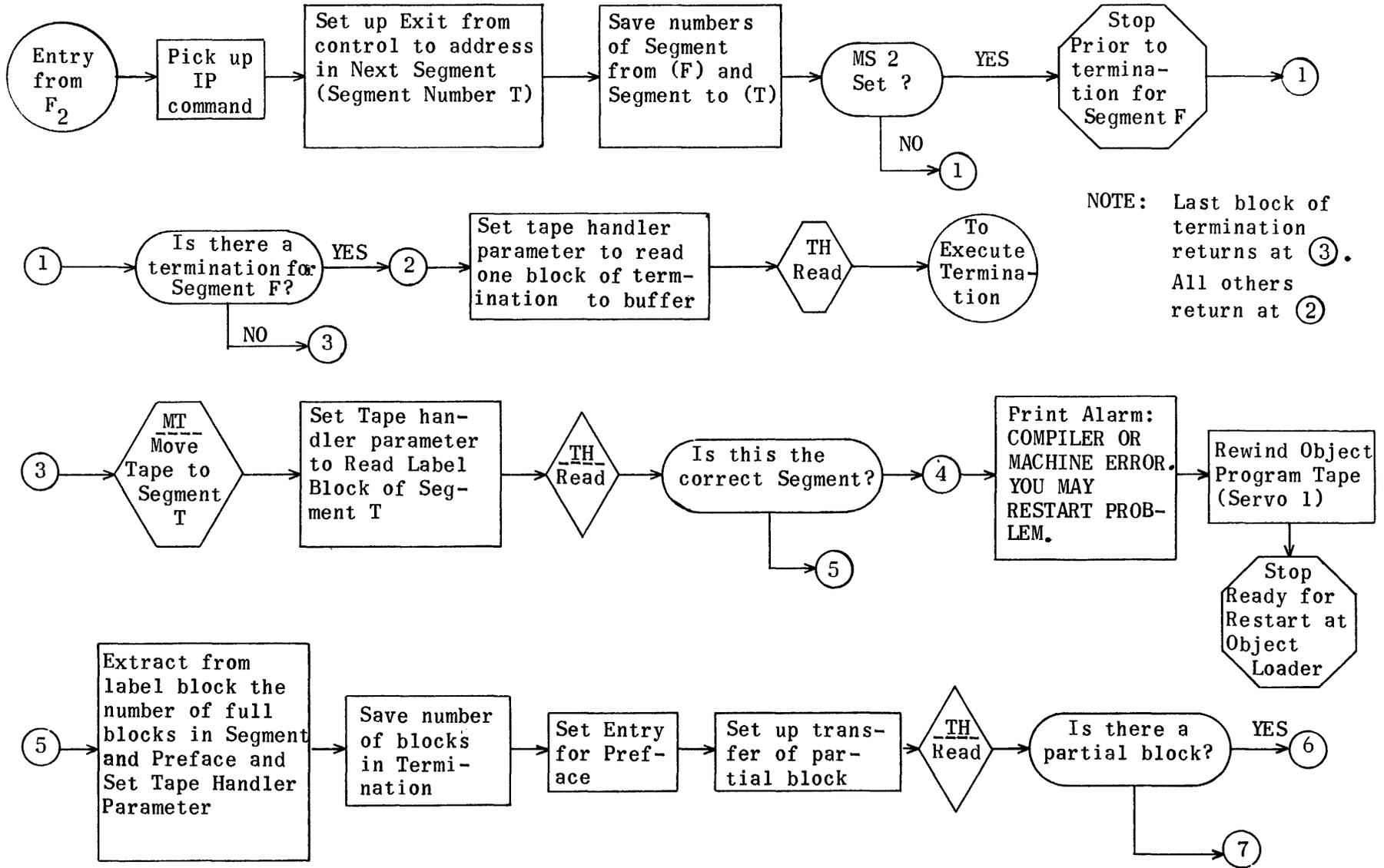
At the time the tape is to be moved from Segment FF to Segment TT, it is positioned exactly at the end of Segment FF. Hence the number of blocks the tape is to be moved to position it at the beginning of Segment TT is:

$$\begin{aligned} \text{Case 1 : } & B(FF + 1) + B(FF + 2) + \dots + B(TT - 1) \\ \text{Case 2 : } & B(TT) + B(TT + 1) + \dots + B(FF) \end{aligned}$$

The tape is moved forward for Case 1, backward for Case 2.

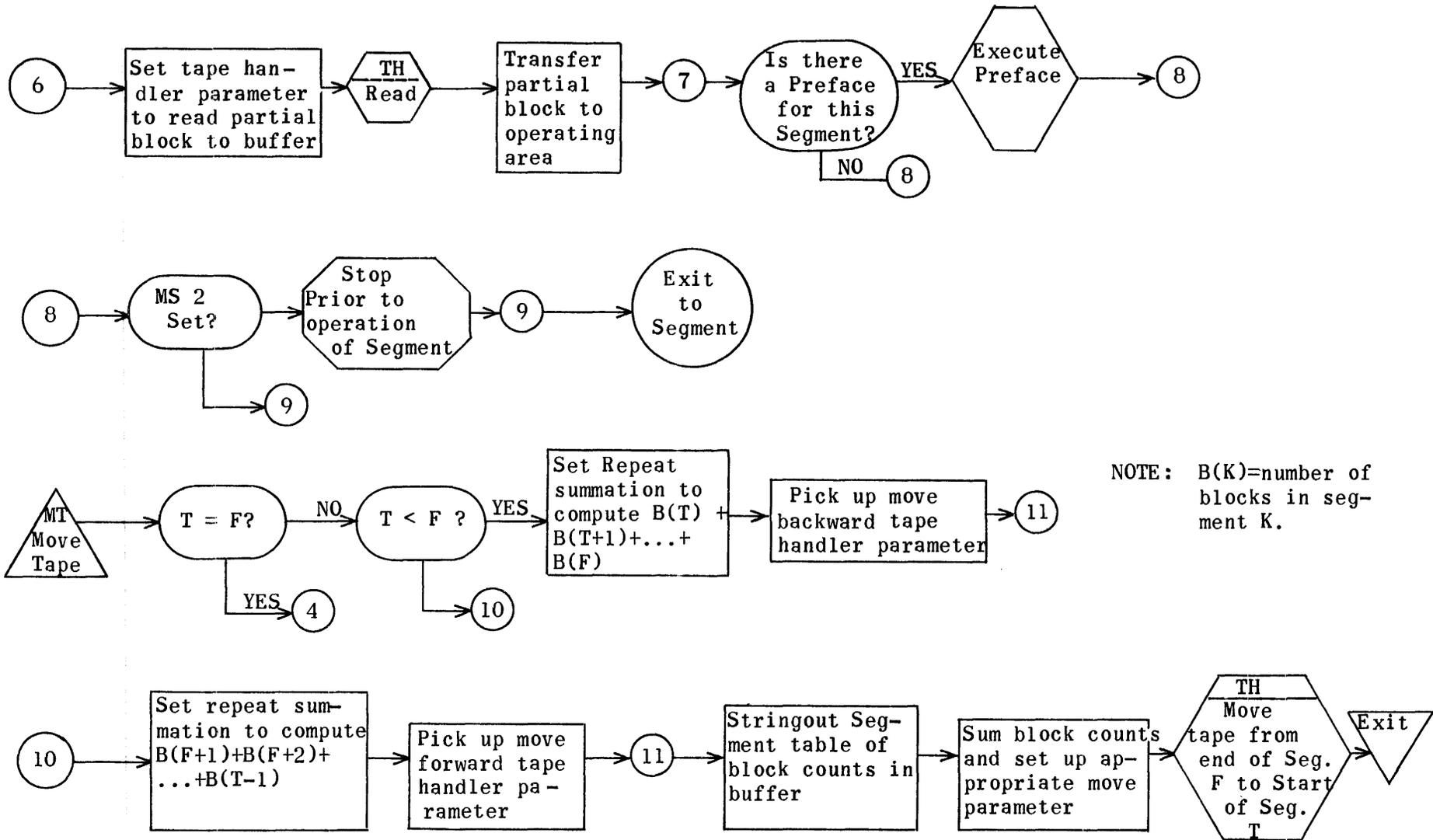
Segment Layout on Object Program Tape





NOTE: Last block of termination returns at ③. All others return at ②

Control Section for Object Program



NOTE: B(K)=number of blocks in segment K.

Regions for UNICODE Control

RE	ON4274	Loading address during Initializa- tion Generation
RE	CT5	Operating address during Object Program
RE	MT77	Move tape routine
RE	KK142	
RE	KT161	
RE	TB166	Segment table
RE	GT210	Tape handler
RE	BU610	Termination buffer
RE	DA77300	Object Program Loader
RE	PR77250	Flex print routine

Object Program Control

	IA	ON		Segment from = F Segment to = T F2 : Jump to control	
	MJ	0	CT		
	0	30000	30000		
	0	30000	30000		
	0	30000	30000		
CTO	TP	0	A	} Set up address of IP command	
1	SS	KK0	17		
2	TU	A	CT3		
3	TP	30000	Q		
4	TV	Q	CT52	IP command → Q Set up exit from Control to segment T	
5	QT	KK13	A	F.221 → A	
6	LT	17	KT1	F → KT1	
7	QT	KK14	A	T.215 → A	
10	LT	25	KT2	T → KT2	
11	MS	20000	CT12	Selective stop at end of segment	
①	12	TP	KT0	A } Is there a Termination for segment F?	
	13	ZJ	CT14	CT17	} Yes, so read block of Termination to buffer and execute. Returns at CT14 or CT17
②	14	TP	KK15	GT3	
15	RJ	GT2	GT0		
	16	MJ	0	BU0	} Move tape to segment T
③	17	RJ	MT0	MT1	
	20	TP	KK15	GT3	} Read label block of segment T
	21	RJ	GT2	GT0	
	22	TP	BU2	A	} Is this segment T?
	23	EJ	KT2	CT25	
	24	MJ	0	CT53	} No, so go to print alarm
⑤	25	TP	BU4	A	
	26	AT	KK3	GT3	} Extract information from label Set up parameter to read full blocks of segment and Preface
	27	TV	TB0	GT3	
	30	TP	BU3	KT0	
	31	TP	BU6	CT50	} Set entry for Preface Set up partial block word count
	32	TP	KK12	KT3	
	33	TU	BU5	KT3	} Set up transfer of partial block
	34	TP	KK16	A	
	35	AT	KT3	CT44	
	36	TV	BU5	CT45	
	37	RJ	GT2	GT0	} Read full blocks of segment and Preface Is there a partial block?
	40	TP	KT3	A	
	41	ZJ	CT42	CT46	} Yes, so read it to buffer
⑥	42	TP	KK15	GT3	
	43	RJ	GT2	GT0	
	44	RP	30000	CT46	} Transfer partial block to operating location
	45	TP	BU0	30000	

⑦	46	TP	KT0	A	}	Is there a Preface for segment T
	47	ZJ	CT50	CT51		
	50	RJ	30000	30000		Execute Preface
⑧	51	MS	10000	CT52		Selective stop before operation of segment
⑨	52	MJ	0	30000		Execute segment T
④	53	TP	CT60	PR3	}	Print alarm
	54	RJ	PR2	PRO		
	55	EF	0	CT57		Rewind Object Program tape (Uniservo 1)
	56	MS	0	DAO		Stop
	57	02	200	10000		
	60	0	CT61	11		Parameter
	61	45	47160	30715		Cr ↑ C O M P
	62	14	11201	20403		I L E R Δ O
	63	12	04073	01605		R Δ M A C H
	64	14	06200	42012		I N E Δ E R
	65	12	03124	52503		R O R Cr Y O
	66	34	04073	02504		U Δ M A Y Δ
	67	12	20240	13012		R E S T A R
	70	01	04151	20323		T Δ P R O B
	71	11	20074	55700		L E M Cr ↓
	MT0	MJ	0	30000		Exit
	1	TP	KT2	A		Entry
	2	ST	KT1	A		T - F → A
	3	ZJ	MT4	CT53		If T = F, go to print alarm
	4	SJ	MT5	MT13		Is T > F?
	5	TN	A	A		No, so F - T → A
	6	SA	KK0	17		(F - T + 1) · 2 ¹⁵ → A
	7	AT	KK4	MT35		Set up repeat summation on Segment table
	10	SP	KT2	17		T · 2 ¹⁵ → A
	11	TP	KK2	KT4		Pick up move back dummy
	12	MJ	0	MT20		
⑩	13	SS	KK0	17		T > F, so form (T - F - 1) · 2 ¹⁵ in A
	14	AT	KK4	MT35		Set up repeat summation
	15	TP	KT1	A		F → A
	16	SA	KK0	17		(F + 1) · 2 ¹⁵ → A
	17	TP	KK1	KT4		Pick up move forward dummy
	20	AT	KK5	MT36		Set to pick up first term
	21	RP	30020	MT23	}	Segment table → buffer
	22	TP	TB1	BU100		
	23	TP	KK7	KT3		Set index
	24	TP	KK10	MT31		
	25	RP	20020	MT27	}	Position columns
	26	LQ	BU100	11		
	27	TP	KK6	Q	}	String out the block counts of the Seg-

30	RP	30020	MT32	J	Mask out columns	} ment Table to simpli- fy the summation
31	QT	BU100	BU0			
32	RA	MT31	KK11			
33	IJ	KT3	MT25		4 columns strung out?	
34	TP	KK12	A			
35	RP	20000	MT37	}	Add block counts to determine the	
36	SA	BU0	0			number of blocks to move tape
37	LA	A	25			
40	AT	KT4	GT3	}	Add sum of blocks to parameter and	
41	RJ	GT2	GT0			move tape to segment T
42	MJ	0	MT0		To exit.	
KK0	0	0	1			
1	30	1	0		Move forward dummy	
2	40	1	0		Move backward dummy	
3	50	1	30000		Read forward dummy	
4	RP	20000	MT37	}	Repeat summation dummies	
5	SA	BU0	0			
6	0	0	777		Segment table column mask	
7	0	0	3			
10	QT	BU100	BU0			
11	0	0	20			
12	0	0	0			
13	0	7700	0		Segment "from" mask	
14	0	77	0		Segment "to" mask	
15	50	101	BU0		Parameter to read one block to buffer	
16	RP	30000	CT46		Partial block repeat dummy.	
KTO	0	0	0		Number of blocks of termination (= 0 for F = 0)	
1	0	0	0		F = segment number "from"	
2	0	0	0		T = segment number "to"	
3	0	0	0			
4	0	0	0			
	CA	ON165				

Object Program Tape Handlers

Since the 1103A and 1105 Tape Handlers which are put on the Object Program Tape by Initialization Generation are the same as those used in the Translation Phase, only their regional assignments are shown here. Flow charting, coding, and an explanation of them may be found in Section III, 3, a— Translation Subroutines.

Object Program Tape Handler Regions

	1103A		1105
	RE TG4461 RE TH210 RE WB244 RE WW256 RE RF270 RE IA300 RE RR301 RE RE321 RE RA330 RE RB367 RE RW377 RE MF404 RE MB415 RE PC417 RE WE440 RE CF451 RE CC464 RE CE524 RE CD547 RE VV557 RE CR565	{ Loading address during Initial- ization Gener- ation }	RE TG4461 RE TH210 RE RW257 RE RF264 RE RB272 RE IA300 RE EX301 RE WB304 RE WW316 RE RR330 RE RE346 RE MF411 RE MB422 RE PC424 RE WE445 RE CC456 RE CE516 RE CF534 RE CD547 RE VV557 RE CR564
Length = 370g words	Operating ad- dresses during Object Program	Length = 365g words	

VI. PROCESSING PHASE

VI PROCESSING PHASE

The Processor uses as input the Op File III for each segment together with the library and generated subroutines with their preludes. From this input the Processor assembles the required subroutines for each segment. As each subroutine is processed, the relatively coded addresses are changed to the proper machine coded operating addresses. Cross reference call words are replaced by the necessary machine coding to accomplish the cross reference, depending on whether the reference is "within a segment" or "from one segment to another". When all the routines for one segment have been processed, the segment together with its Preface and Termination is transferred to Uniservo tape. This tape, containing all the segments in sequence, is the Object Program tape. A more explicit description of the methods used in modifying the relative coding follows.

In the initial stage of the Processor the Op File III for the segment to be processed is read from tape into High Speed Storage. When this transfer has been completed, the first subroutine is read from the Generated Routines Tape into the Tape Image in High Speed Storage. At this point the tape handling is temporarily suspended and the actual processing begun. The call word for the subroutine is checked against those listed in Op File III to determine if the subroutine is referenced in this particular segment. The word following the call word is then checked to see if it has a flag indicating a cross reference to another segment. If the call word is listed in the Op File III and is not flagged, the subroutine will be processed at this time. If the subroutine is not to be processed at this time, the next subroutine will be read into the Tape Image and the foregoing procedure repeated. When all the generated routines in the segment have been processed, the Fixed Library

and Standard Library routines are processed in like manner.

The first line to be processed in all cases is the entrance line of the subroutine. Following the modification of this line, each line subject to address modification is processed in order, beginning with the line indicated by the line count of the Tape Image. Each relative address is processed depending on the nature of the coding, to obtain the proper machine coded address.

All addresses within the range 01000 through 07777 are modified as addresses coded relative to 01000; hence, the corresponding absolute address is obtained by subtracting 01000 from the relative address and adding the High Speed Storage operating address for the subroutine in which the address appears. The High Speed Storage operating address for the routine is obtained from the word following the call word for the routine in Op File III for the segment. All other addresses to be modified are in the form of call words (see call word section).

Call words of the form 10xxx, 20xxx, 60xxx, and 70xxx are unique only within the routine in which they appear. The absolute addresses corresponding to such call words are obtained by adding the last three digits of the call word to the initial High Speed Storage operating address of the constant or temporary region associated with the call word. These initial addresses are calculated from information in the Prelude of the routine and provided as inputs to the Address Modification Subroutine.

Call words of the form 61xxx, 63xxx, and 76xxx are modified to obtain the corresponding absolute address, by adding the last two digits of the call word to the initial High Speed Storage operating address of the Pseudo Operation Input Region. The initial address for the Pseudo Operation Input

Region is that of the thirteenth word of the Termination Buffer, and is stored as a constant in this phase.

Absolute addresses corresponding to call words of the form 62xxx and 75xxx are obtained by adding the last two digits of the call word to the initial High Speed Storage operating address of the Function Input Region. The initial address of this Function Input Region is that of the first word of the Termination Buffer and is also stored as a constant in this phase.

Call words of the form 64xxx, 65xxx, or 66xxx are modified to obtain the corresponding absolute address, by adding the last three digits of the call word to the initial High Speed Storage operating address for the non-subscripted variables of the Object Program. This initial non-subscripted variable address is obtained from fixed location 00007.

Similarly, call words of the form 67xxx are modified to obtain the corresponding absolute address by adding the last three digits of the call word to the initial High Speed Storage operating address of the Constant Pool for the Object Program. This initial Constant Pool address is obtained from fixed location 00010.

Call words of the form 71xxx are used to reference absolute addresses in the range 01000 to 01777 and are modified to obtain the absolute address by subtracting 70000 from the call word.

Those call words which reference another routine are of the form 22xxx, 23xxx, 24xxx, 25xxx, 26xxx, 27xxx, 4xxxx, 5xxxx and those which reference a subscripted variable data array are of the form 77xxx. All such call words are considered to be cross-references of the routine, if they appear as addresses to be modified, and must be in Op File III for the segment. If they are not, ALARM 11. COMPILATION INCONSISTENCY (etc.), is typed on the

Flexowriter. With one exception, instructions containing call words of this type are modified by replacing the call word by the High Speed Storage running address of the referenced subroutine or data array. This running address is obtained from the word following the call word in Op File III. The one exception in which this method of modifying a cross reference does not apply is that in which the cross reference is to another segment. Due to restrictions imposed in this system of coding, a reference to another segment occurs only as a one way unconditional jump and is modified by replacing the entire line of coding by an interpret instruction designed to furnish the Control Section with the information necessary to accomplish the desired cross references. This interpret instruction is obtained from the word following the call word in Op File III. It contains the segment number from which the jump is made, the segment number to which the jump is made, and the High Speed Storage running address in the latter segment. When a reference is made to a line of another subroutine other than the entrance line, the line to be modified contains the call word of the referenced subroutine.

When a reference is made to a line in another subroutine other than the first line, the instruction in which the reference is made contains the call word of the referenced routine. This instruction is followed by a special line of coding of the form 10-xxxxx-xxxxx, called a "ten" line. This "ten" line contains the number of the referenced line relative to the first line of the referenced routine. This number will be in the same portion of the "ten" line, i.e., "u" or "v" address, as the call word in the referencing instruction. In processing a reference of this type, the call word is modified as previously mentioned, to obtain in the referencing instruction, the High Speed operating address of the first line of the referenced routine. After both

addresses of this instruction have been modified, the contents of the "ten" line, less the op. code, are added to the instruction to change the High Speed Storage address(es) from that of the first line of the referenced routine to that of the referenced line within the routine.

As the lines of a routine are modified, they are accumulated in the Tape Image and transferred in groups to locations in the Segment Image on drum, corresponding to their High Speed Storage locations during the running of the segment in the Object Program. When all the lines subject to address modification in the routine, i.e., instructions and relative constants, have been processed, the fixed (unmodifiable) constants for the routine are transferred to consecutive locations in the Segment Image, following the last modified line of the routine. Words of zeros, equal in number to the temporary storage locations required by the routine, follow these constants in the Segment Image.

Each generated subroutine and library routine required for the particular segment is processed in this manner. When all the required routines for a segment have been assembled and processed, the entire Segment Image load, including the proper Preface, Termination, and segment label block, is transferred to the output tape to form a segment of the final running program. The Generated Routines Tape is then rewound and the UNICODE System Tape and Standard Library Tapes are moved back to the beginning of the Fixed Library and Standard Library, respectively. The processing of the next segment is then begun.

Each succeeding segment is processed in exactly the same way until all the segments have been processed and written on the output tape. This tape, containing all the segments of the final running program, is then the Object

Program Tape.

In addition, during the execution of this phase, the Sentence Number List is built and stored on drum for use by the Program Listing Phase.

(See Program Listing for format of this list.)

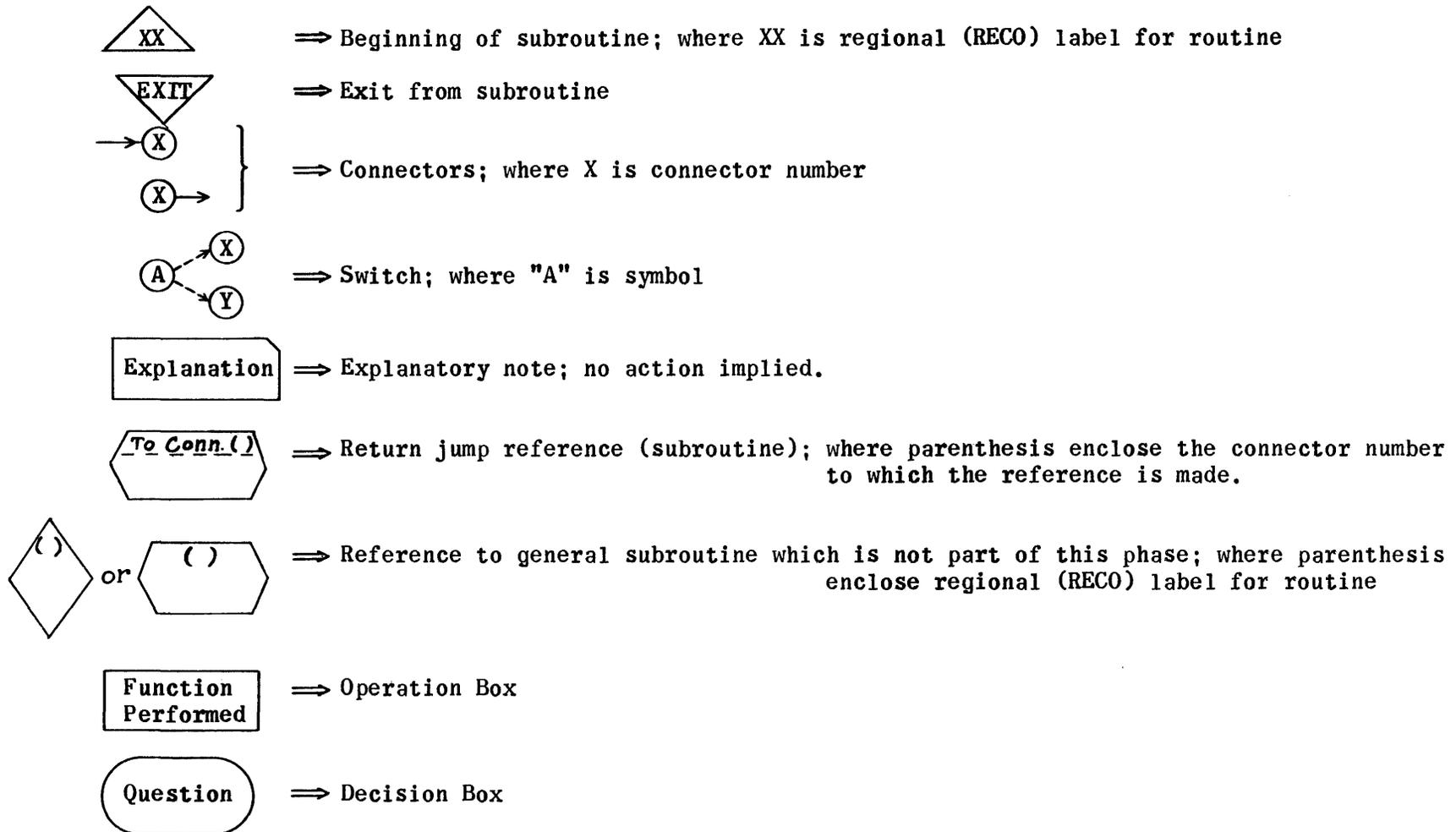
PROCESSOR SETUP BLOCK

Regional Assignments

RE	TH21	Tape Handler
RE	UP421	Uniprint Routine
RE	CK653	Processor
RE	PS7230	Processor Setup Block

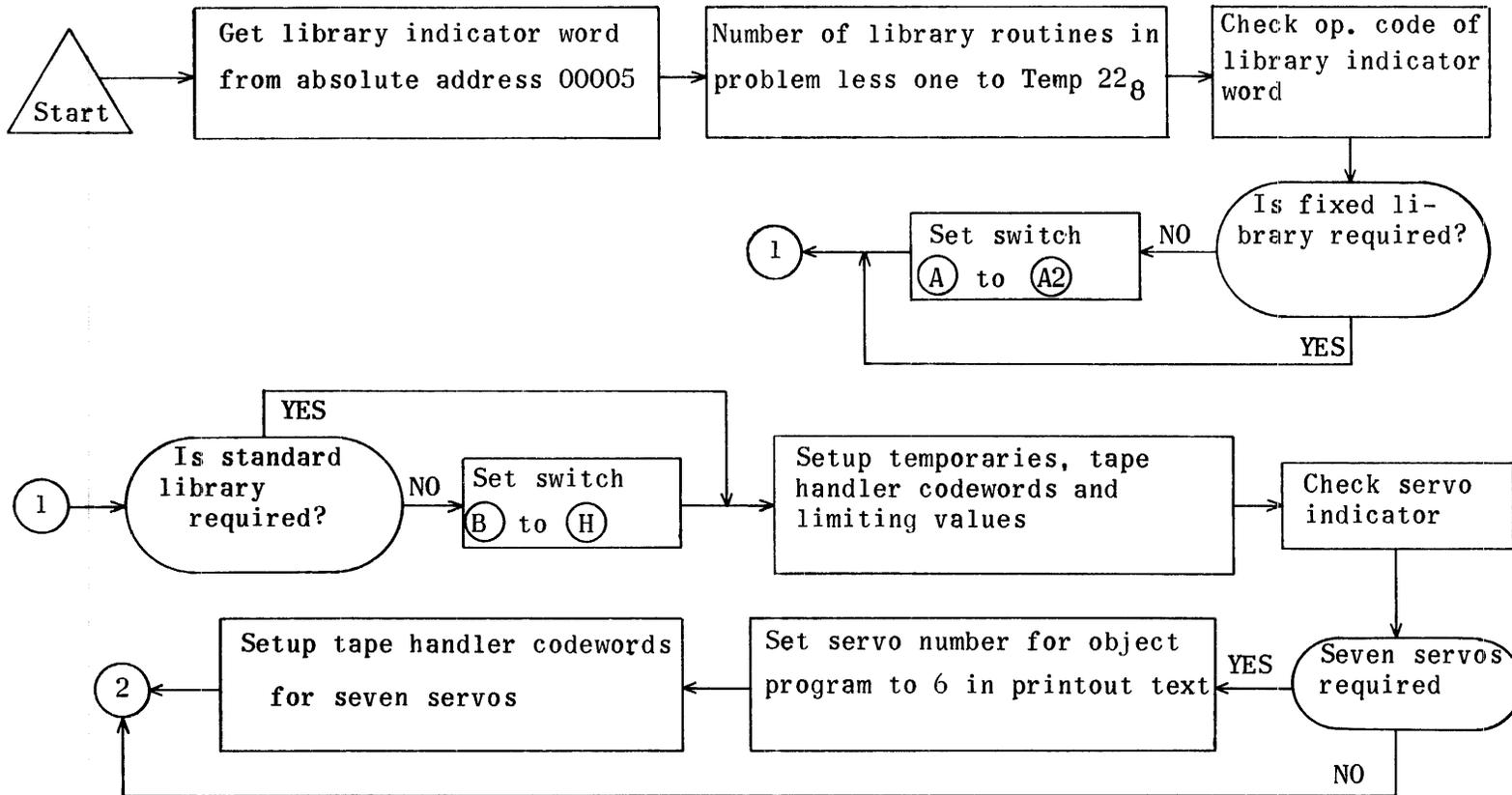
Processor Setup Routine

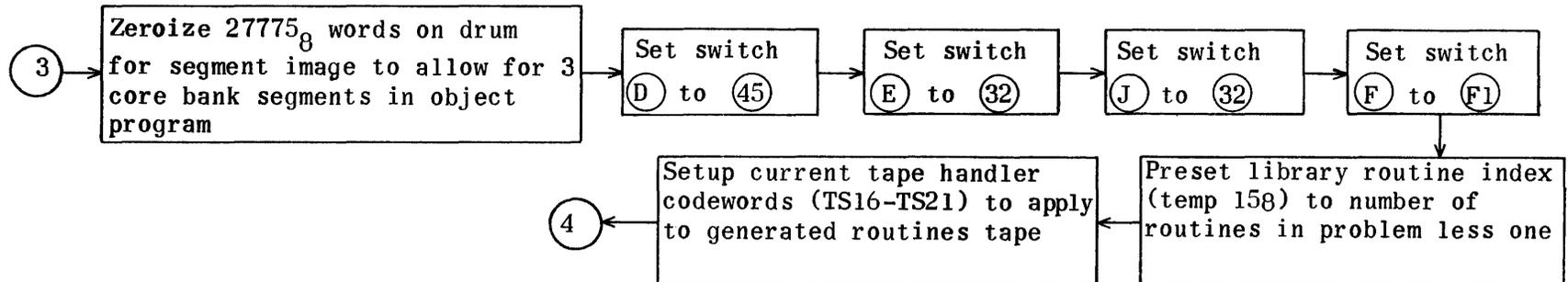
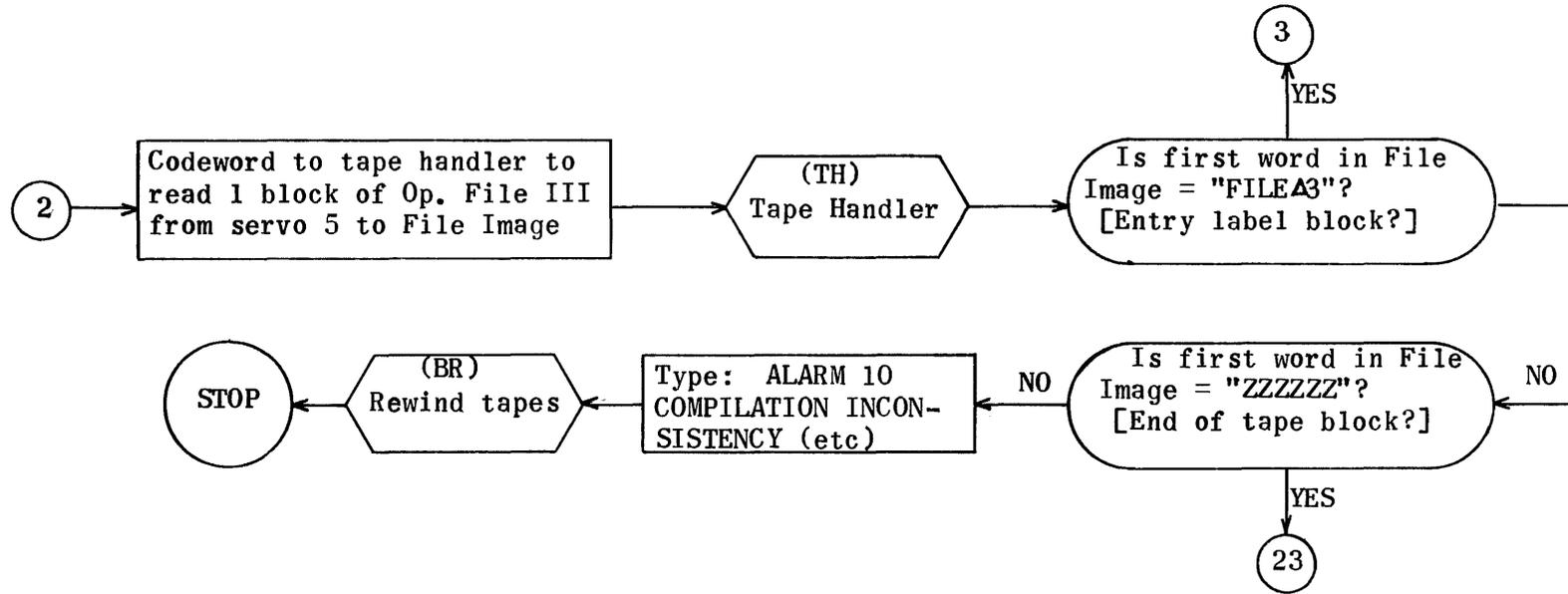
0	IA	PS		
	TP	15	6	Modified Dimension List length to fixed location 00006.
1	TP	PS26	TH3	Parameter to Tape Handler
2	RJ	TH2	TH	Read Processor from Unicode System Tape to core.
3	TP	5	Q	Library indicators → Q.
4	QJ	PS5	PS5	Ignore Fixed Library indicator.
5	QJ	PS6	PS10	Is Standard Library required?
6	TP	PS27	TH3	Yes; parameter to Tape Handler.
7	RJ	TH2	TH	Move Library Tape backward one block.
10	TP	PS13	UP3	Parameter to Uniprint routine
11	RJ	UP2	UP	Type: PASS IV. PROCESSING AND ADDRESS MODIFICATION.
12	MJ	0	CK1	Jump to Processor.
13	00	PS14	12	Parameter for typeout.
14	01	01010	10101	Δ Δ Δ Δ Δ Δ
15	52	24656	50134	P A S S Δ I
16	70	22010	10101	V . Δ Δ Δ Δ
17	01	52545	12630	Δ P R O C E
20	65	65345	03201	S S I N G Δ
21	24	50270	12427	A N D Δ A D
22	27	54306	56501	D R E S S Δ
23	47	51273	43134	M O D I F I
24	26	24663	45150	C A T I O N
25	22	77777	77777	. 77 77 77 77 77
26	50	00601	CK	Parameter to read forward 6 blocks from Uniservo 1
27	40	00102	0	Parameter to move backward 1 block on Uniservo 2
	CA	PS30		

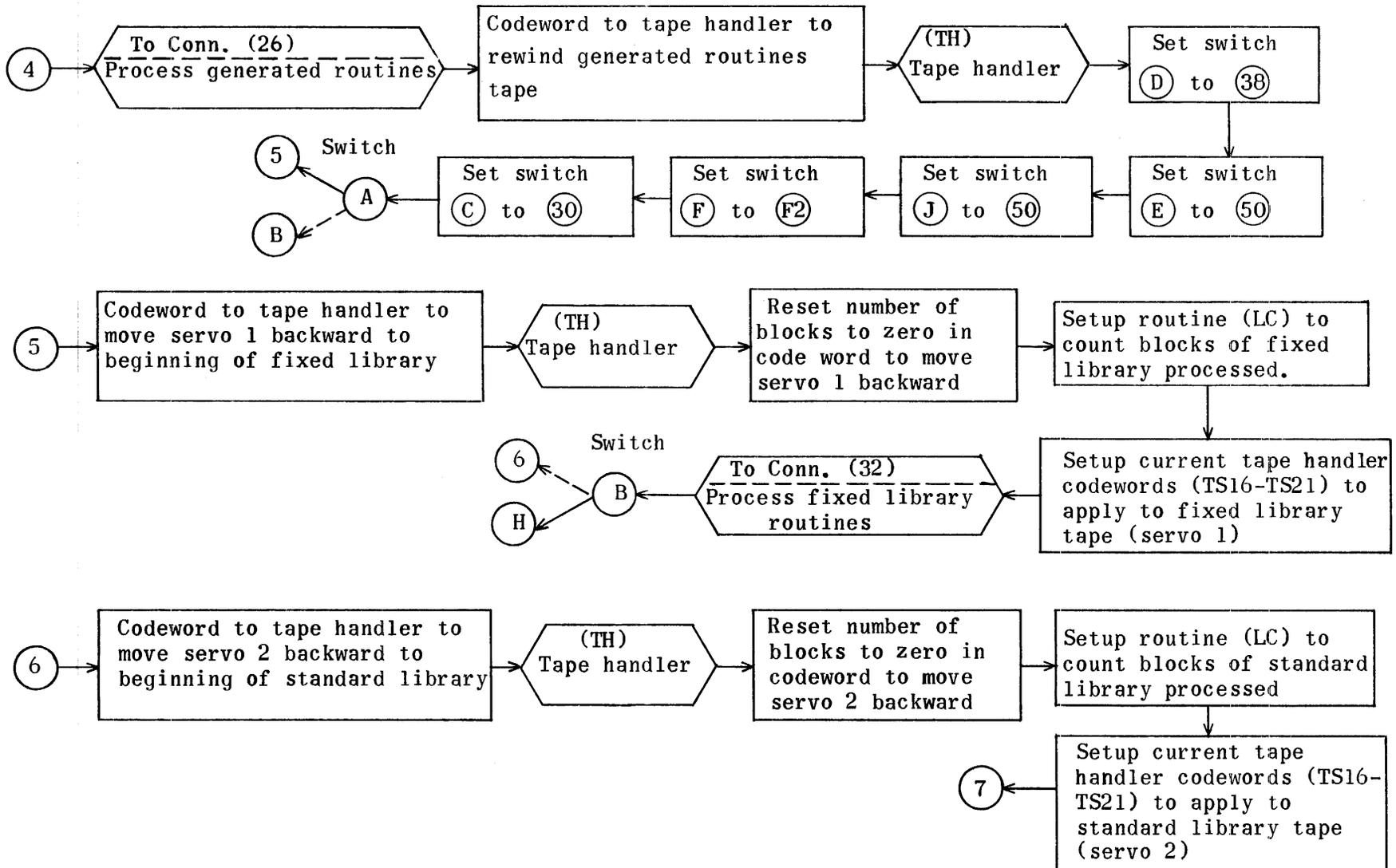


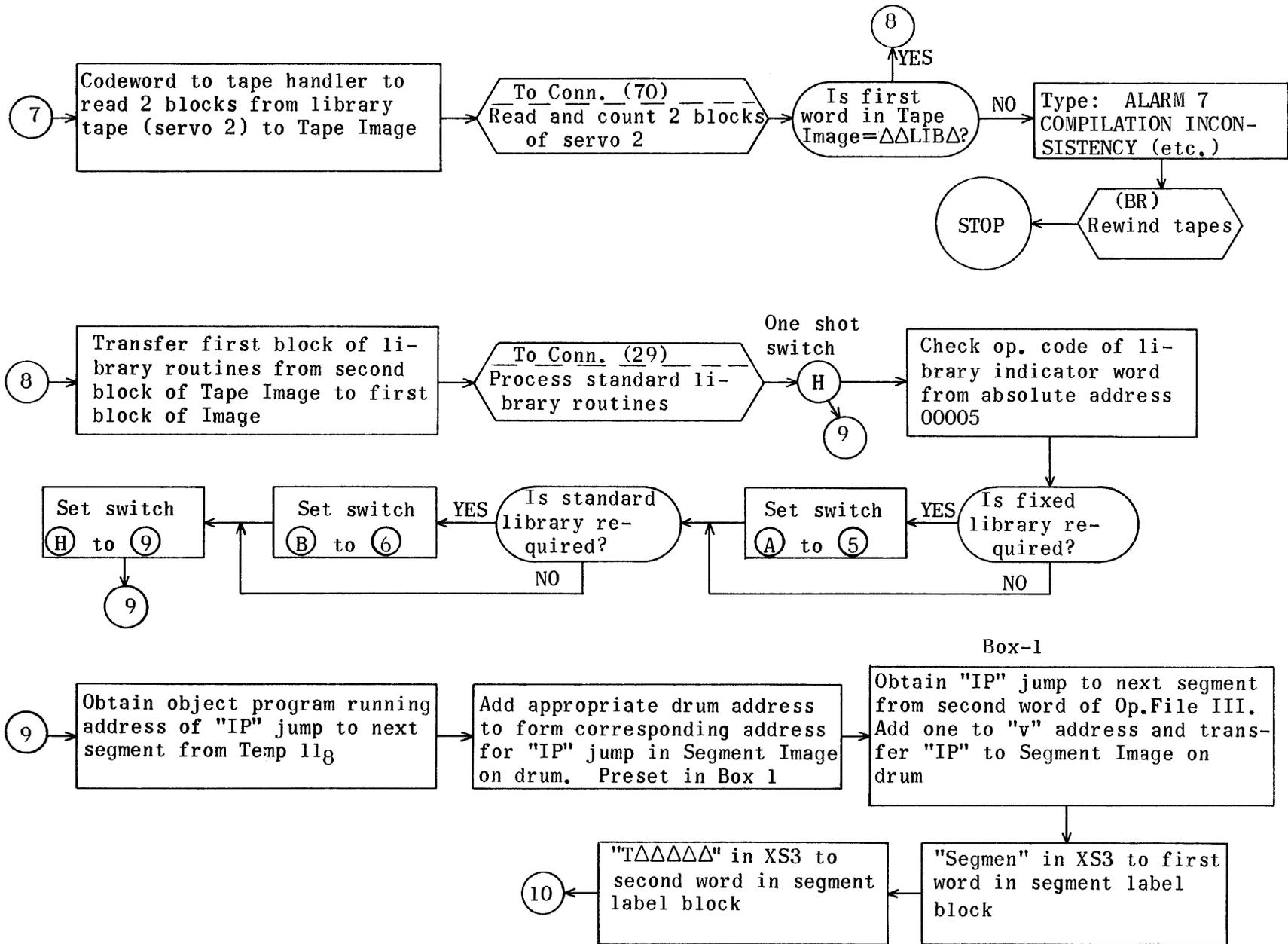
Key to Processor Flow Charts

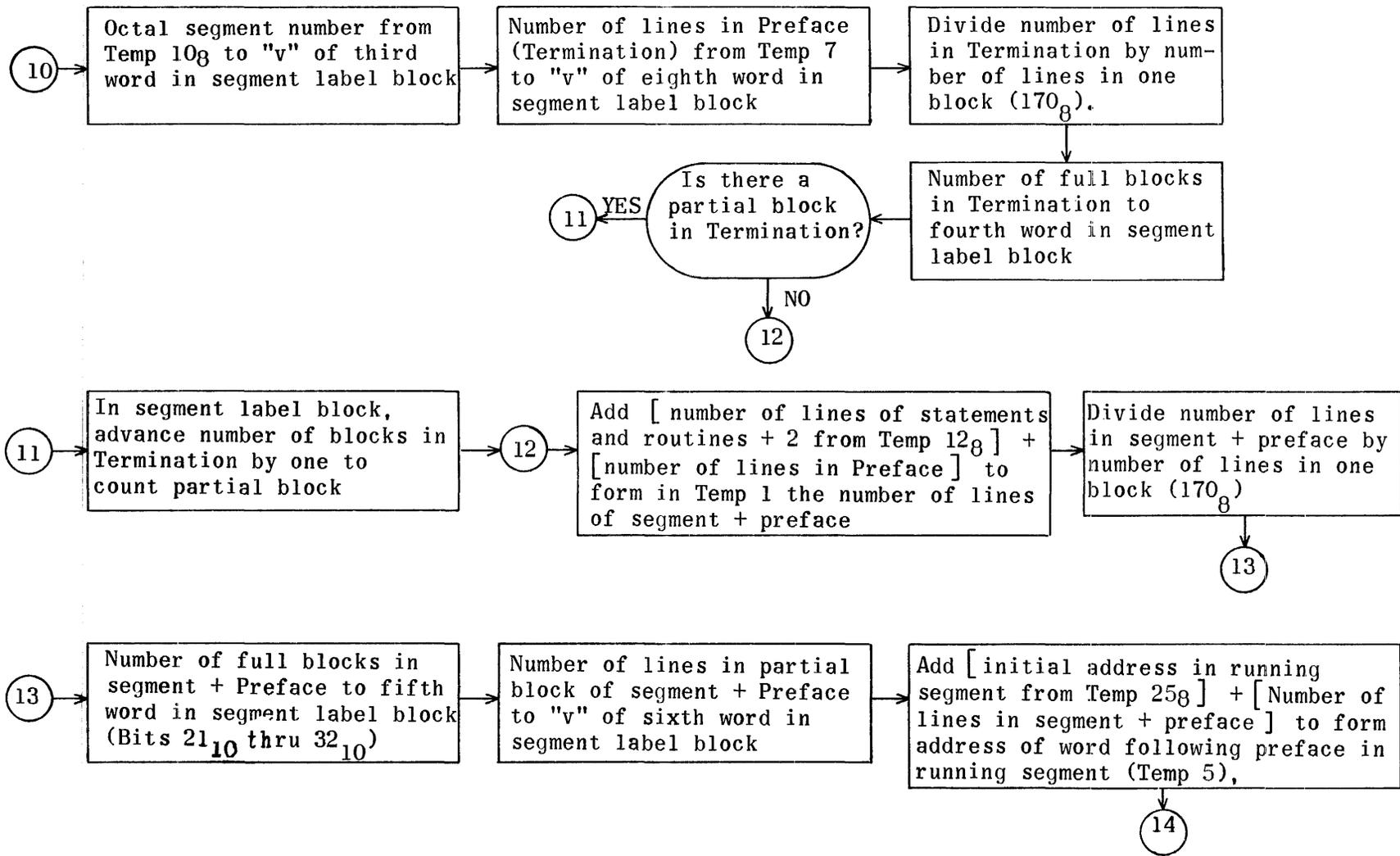
Processor Flow Chart

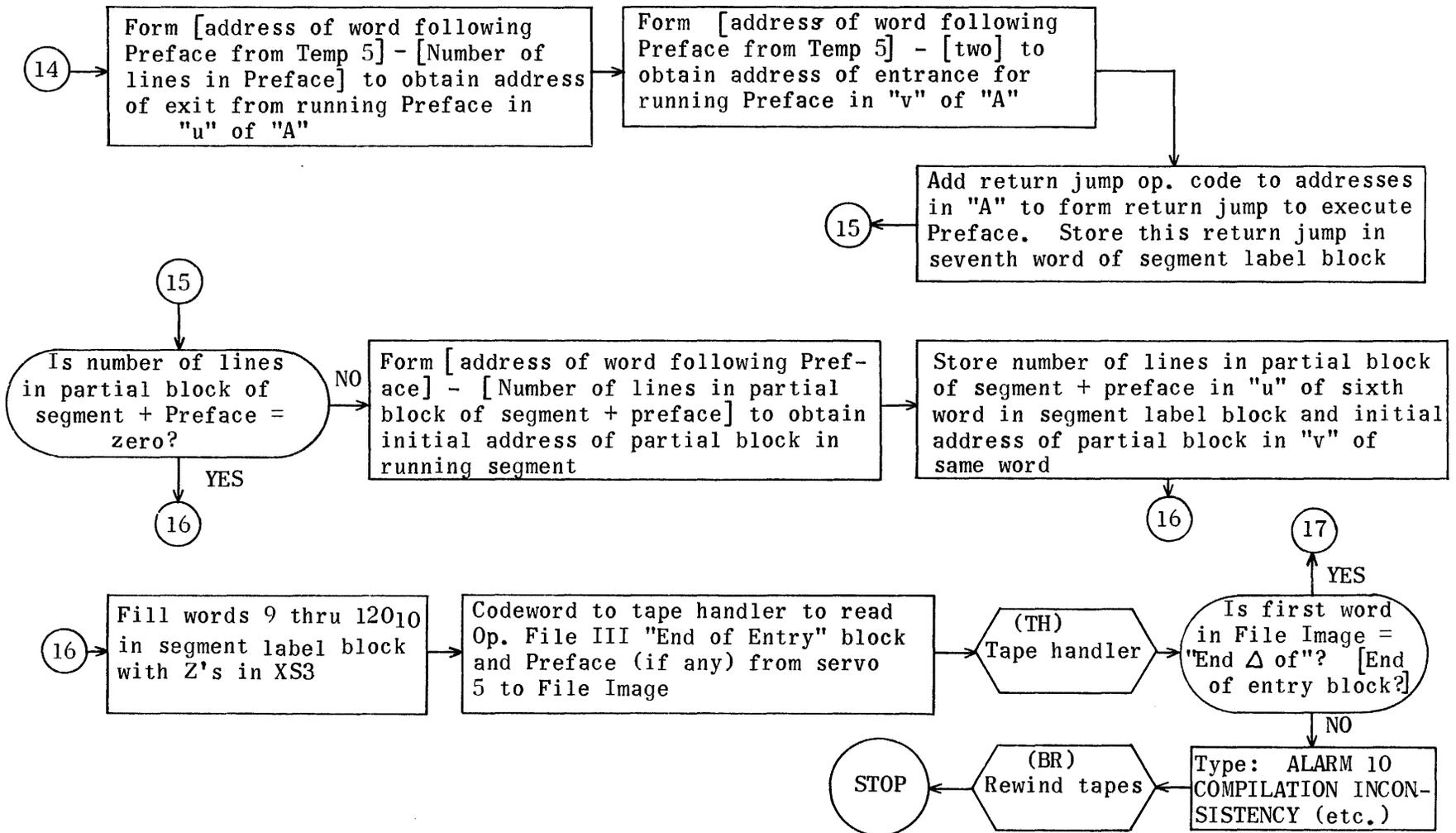


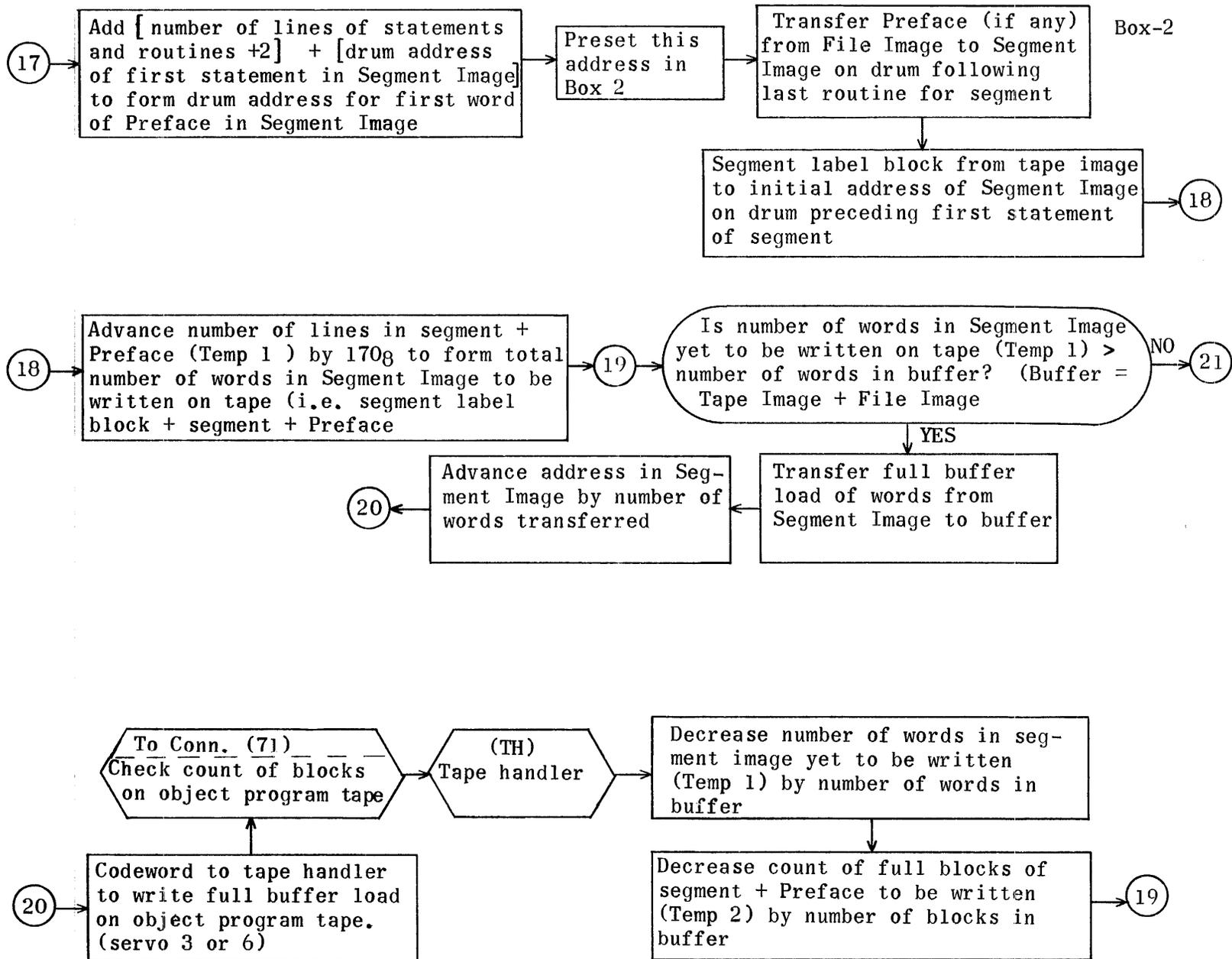


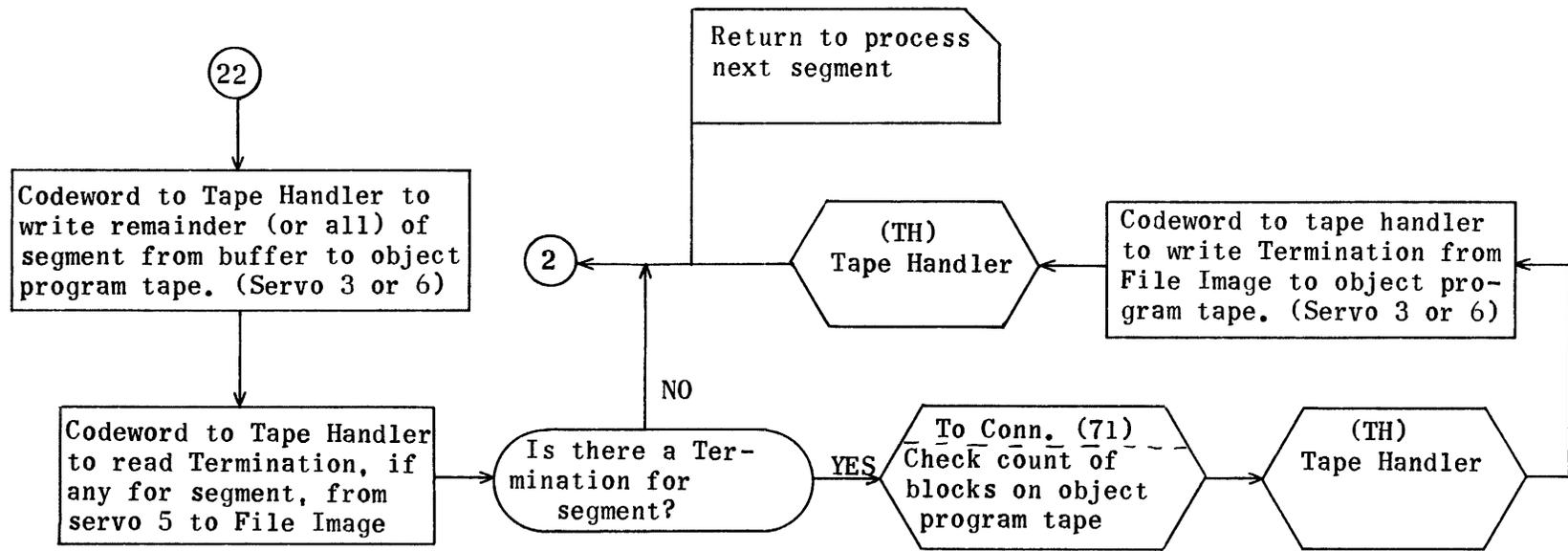
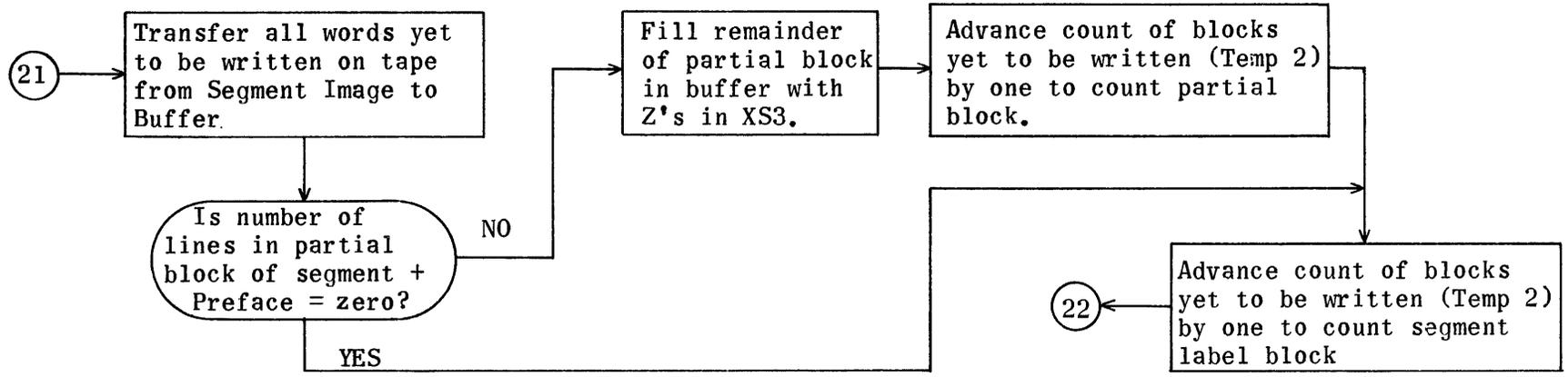


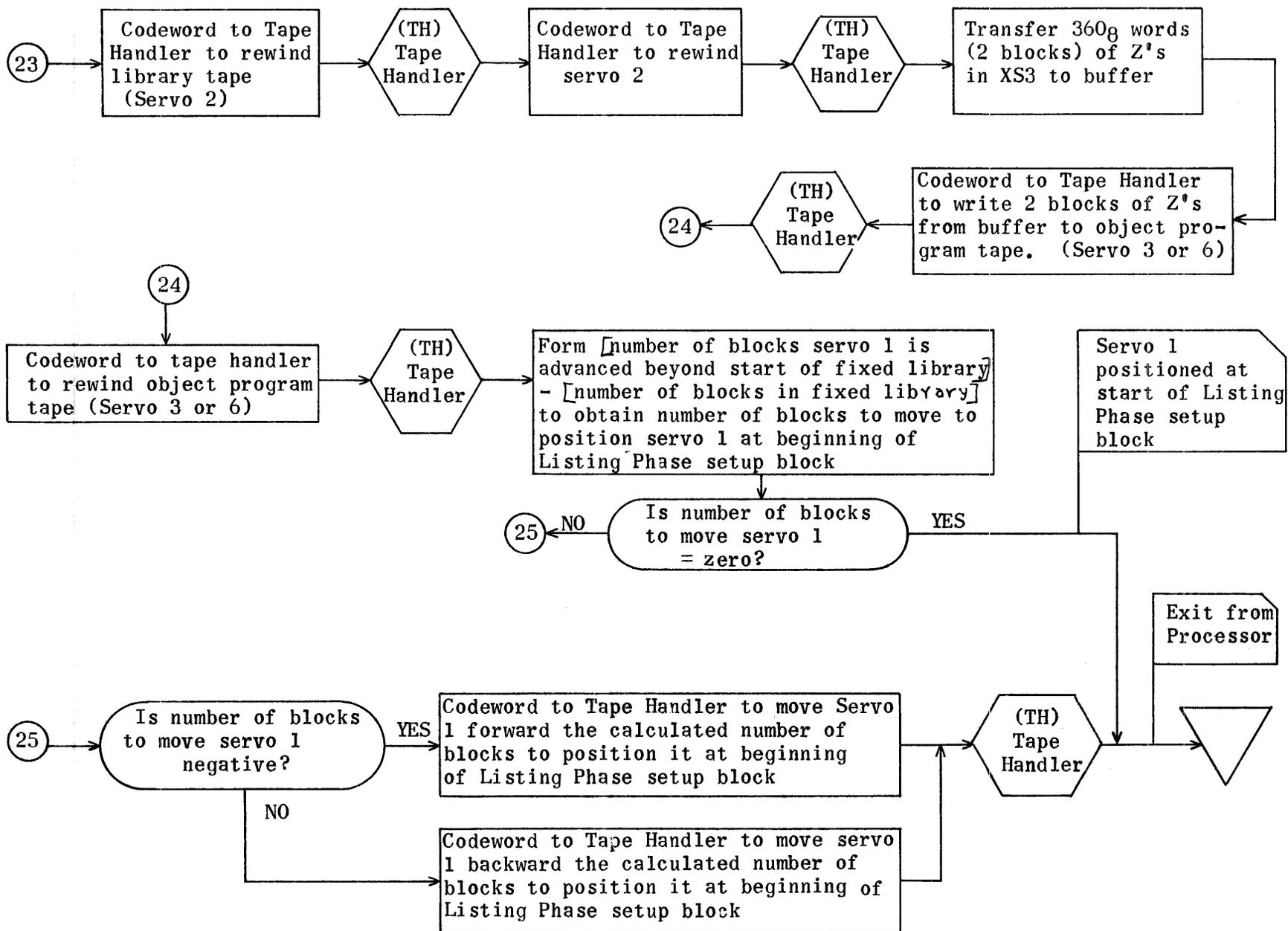


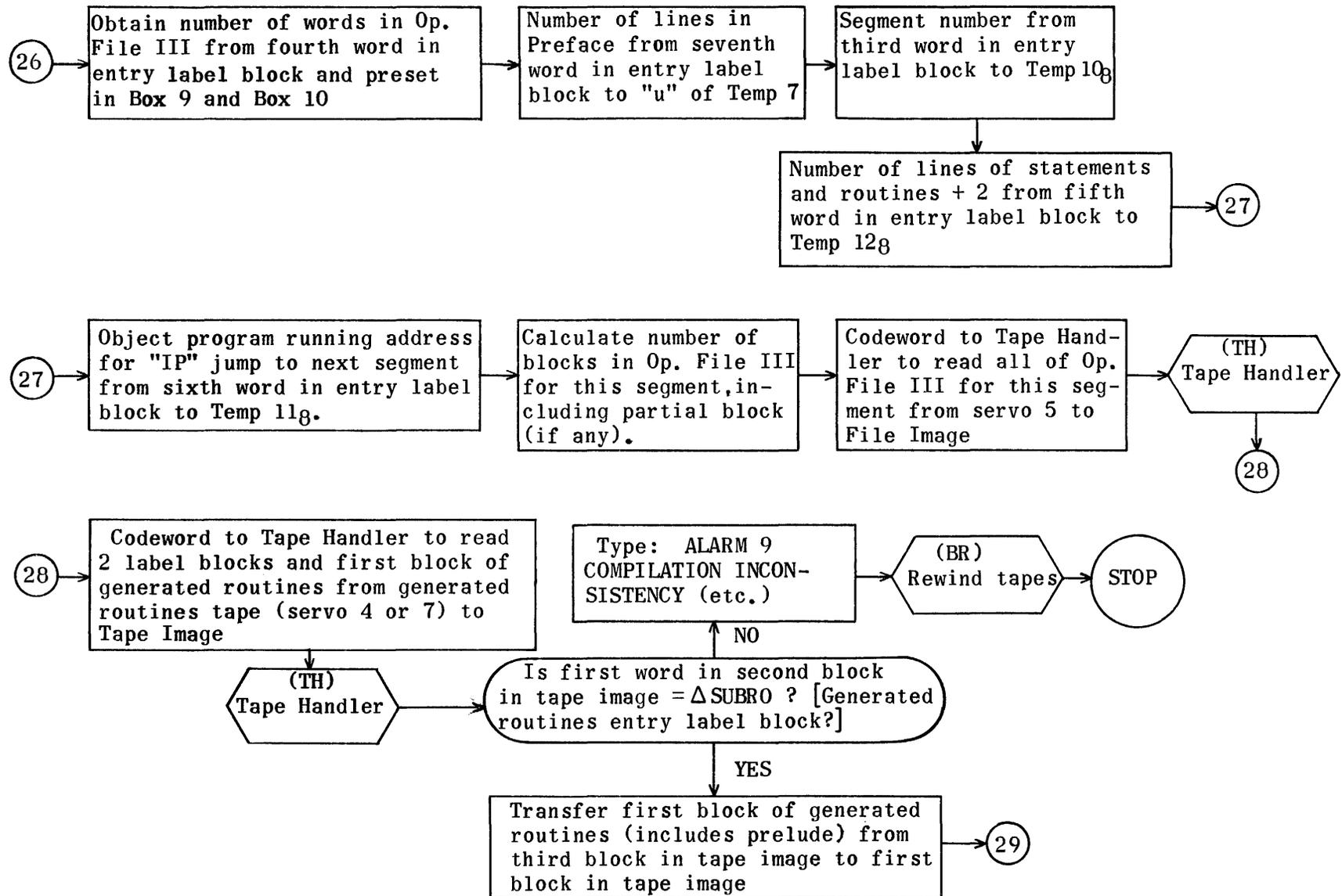


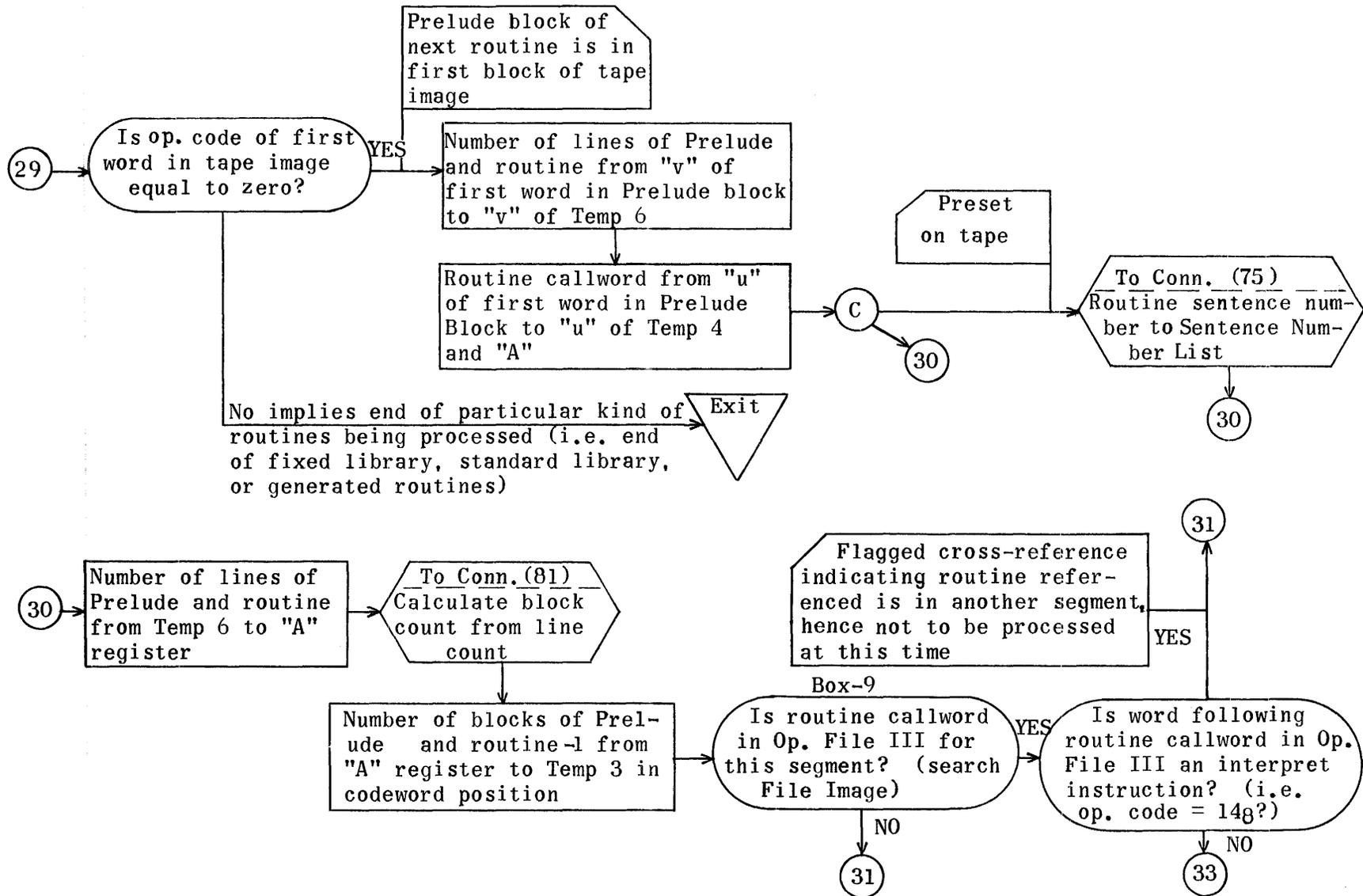


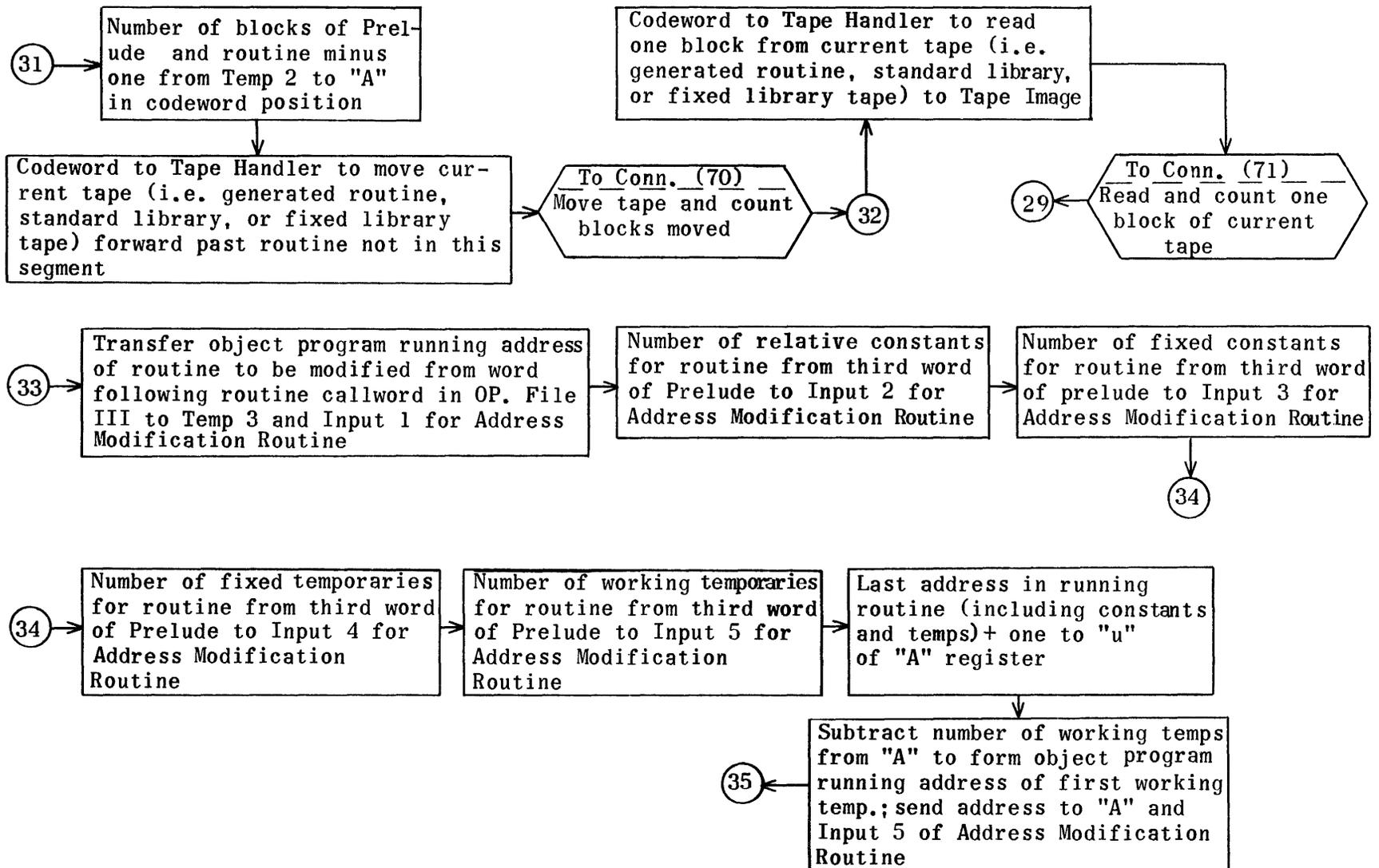


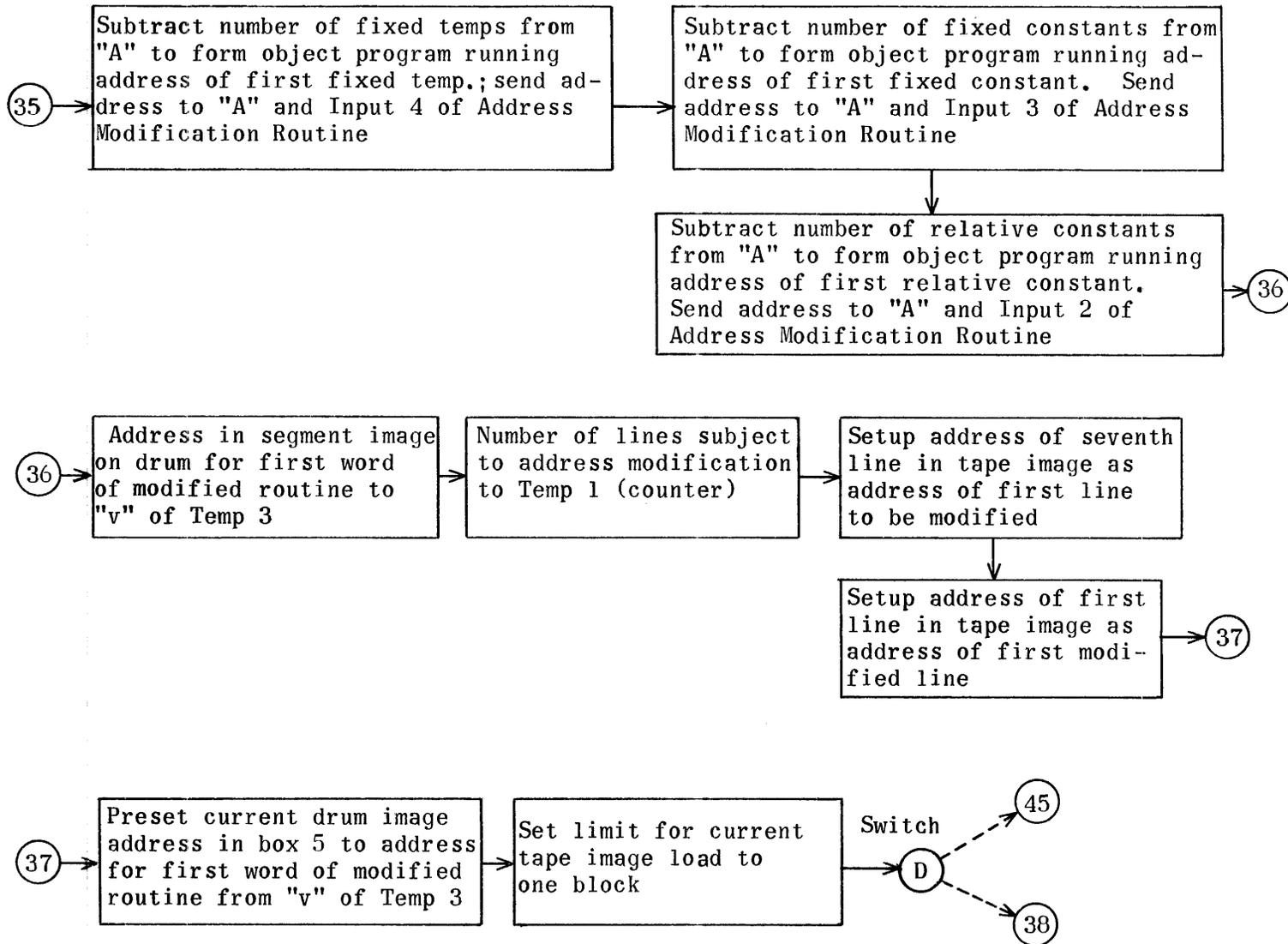


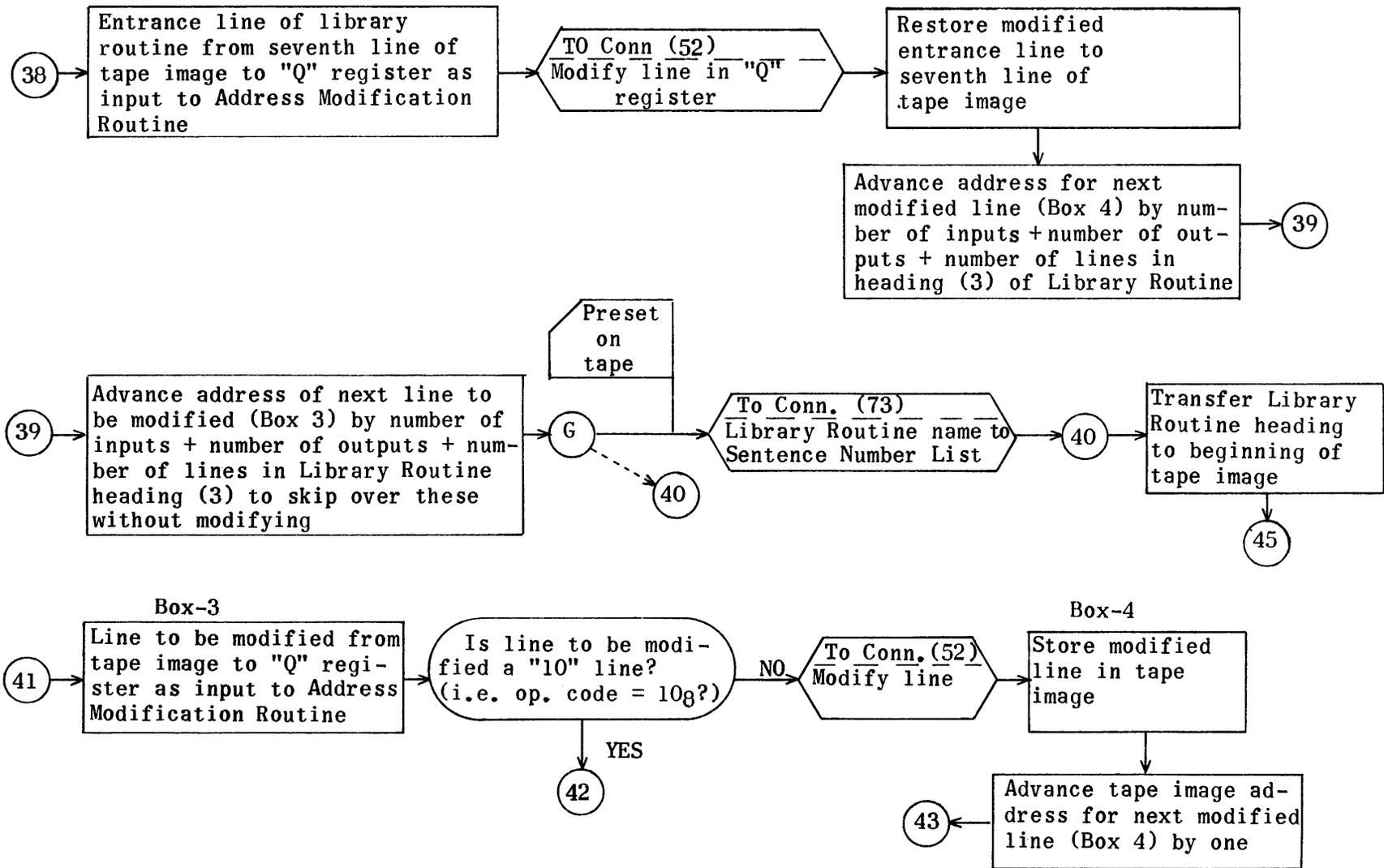


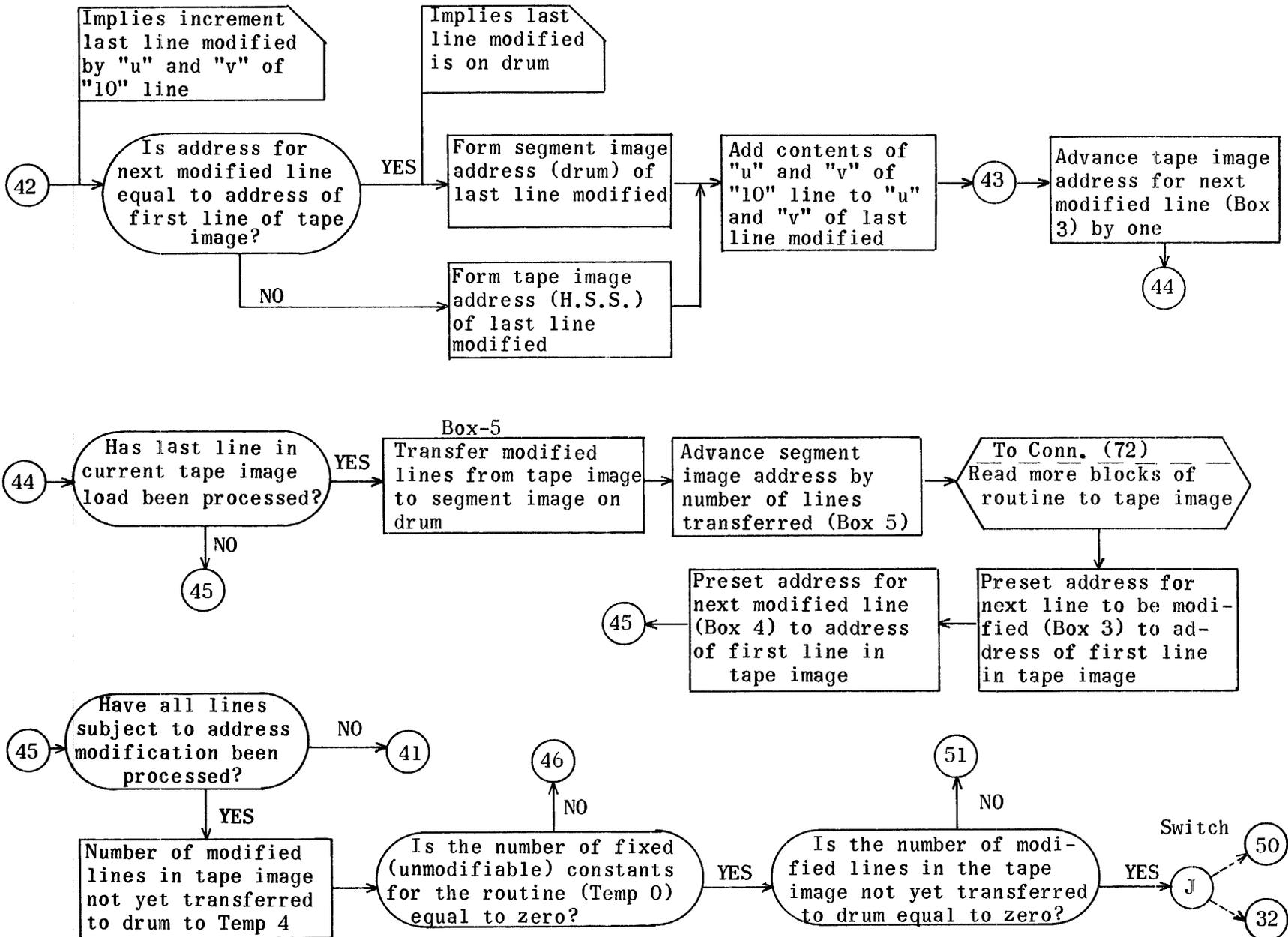


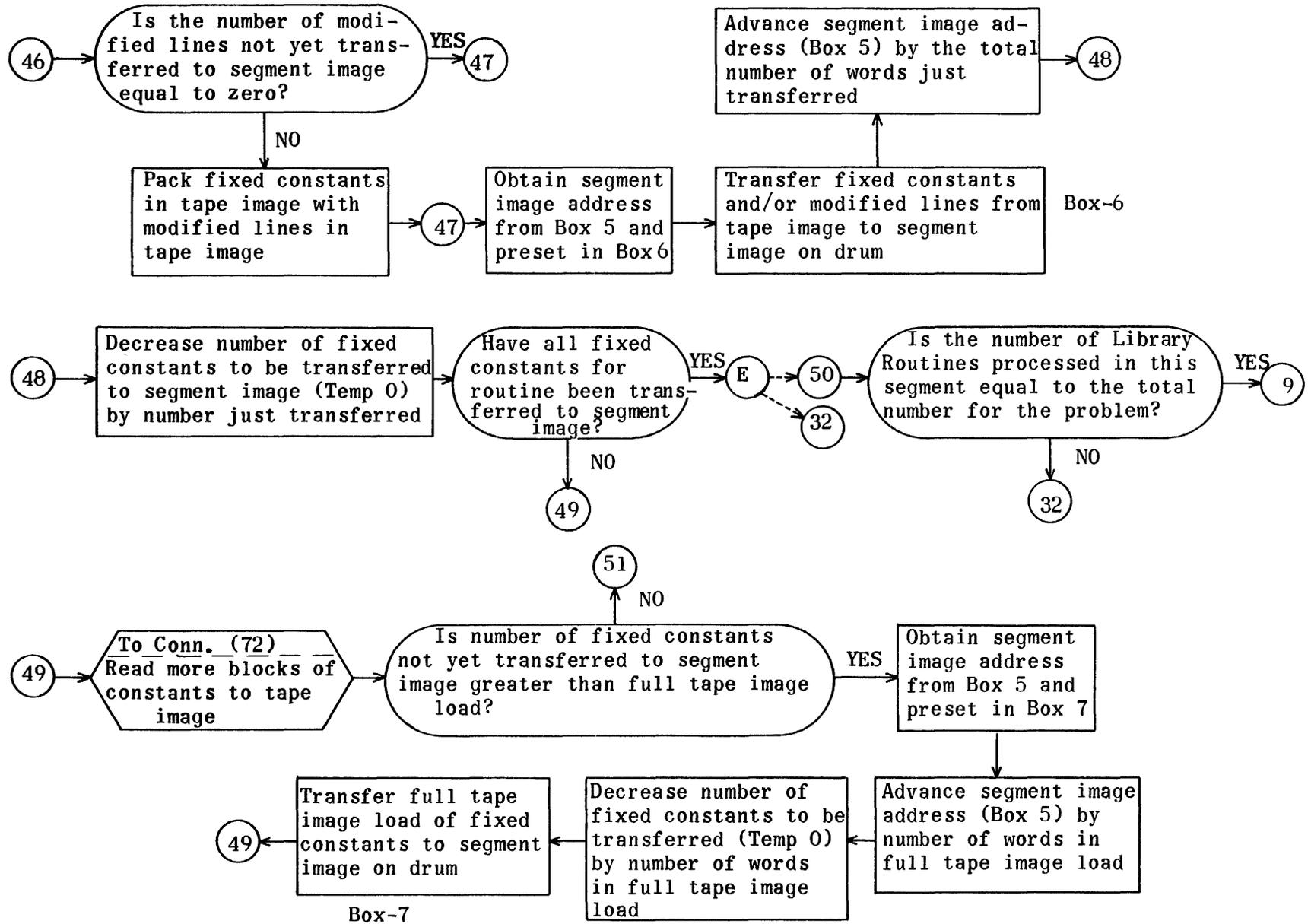


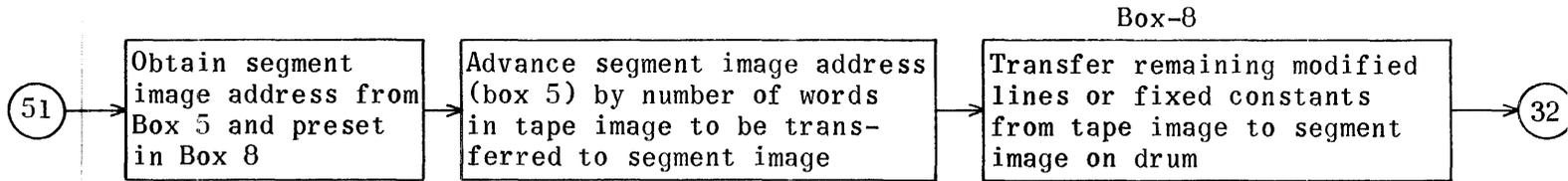




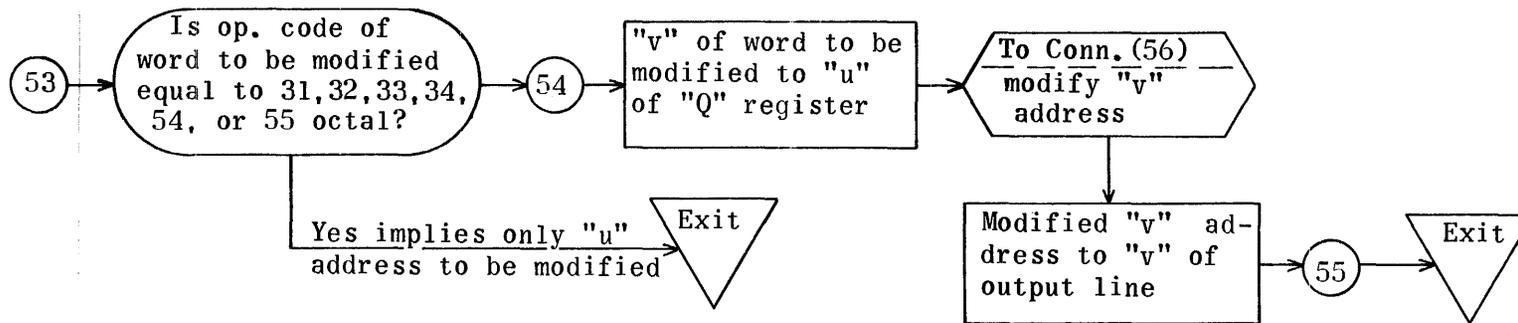
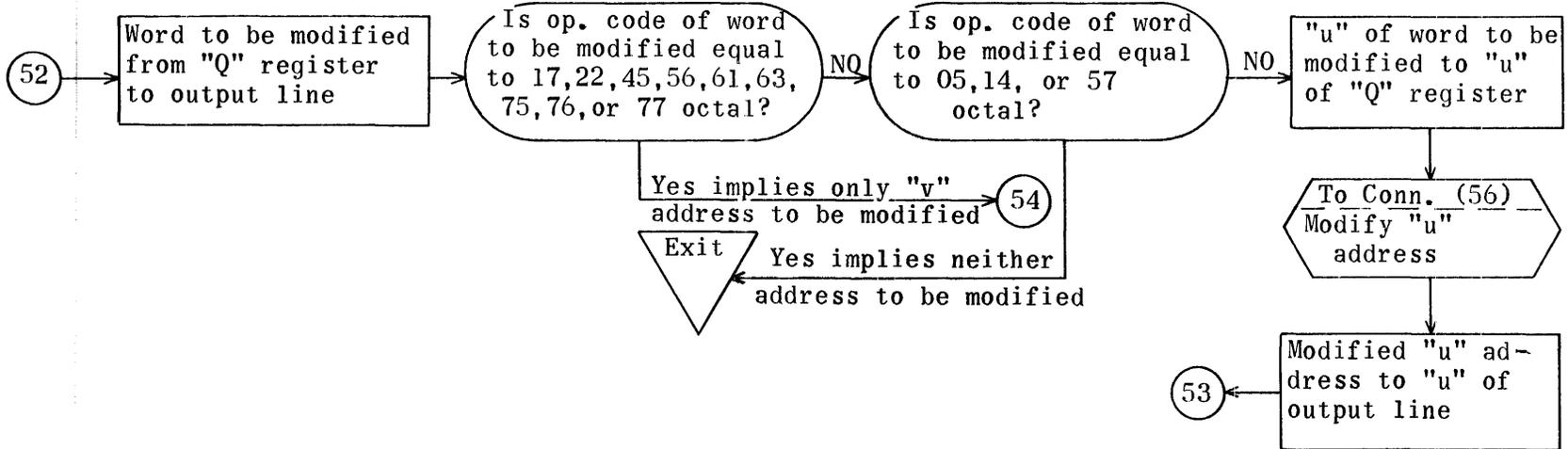






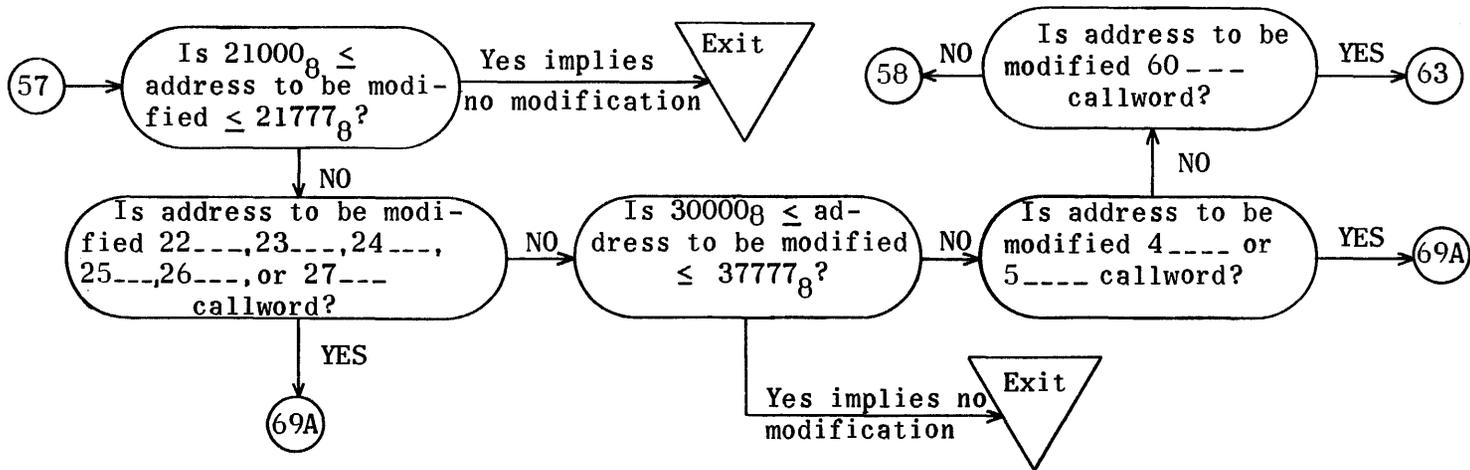
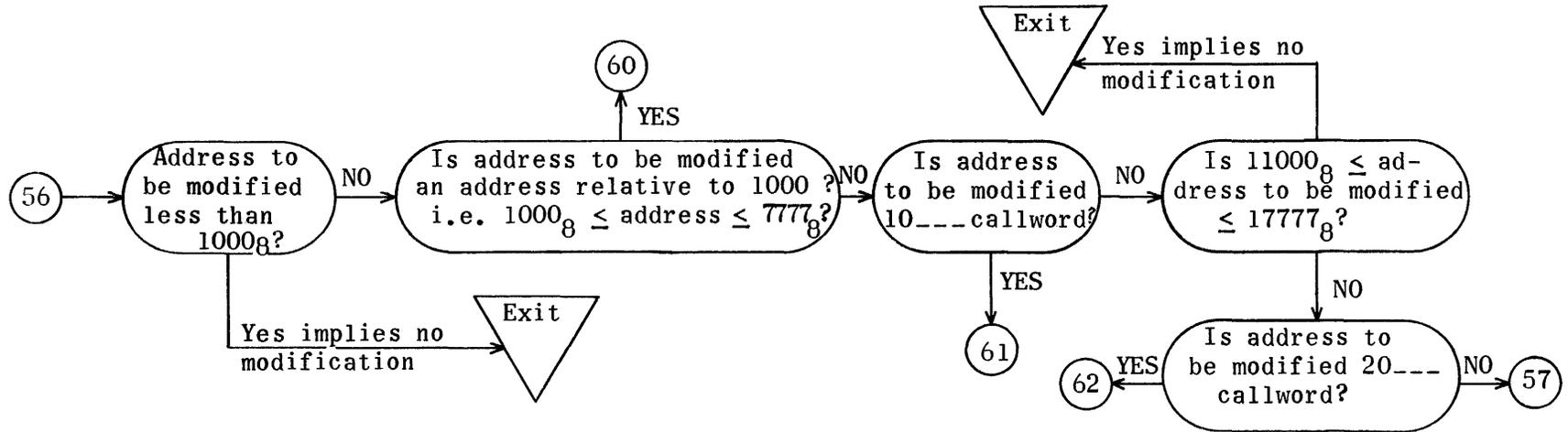


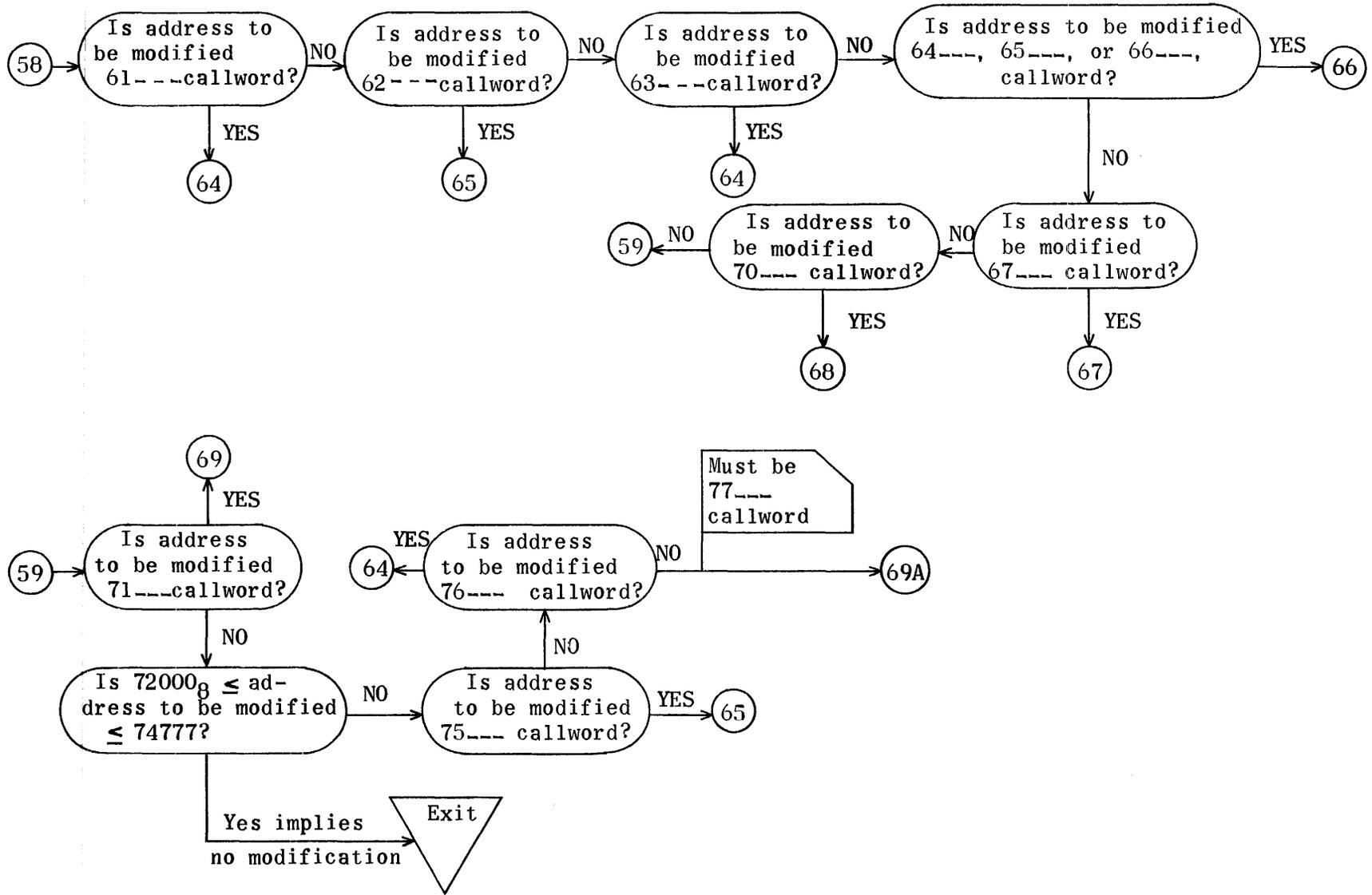
(MR) Address Modification Control Subroutine
 Input = Word to be modified in "Q" Register

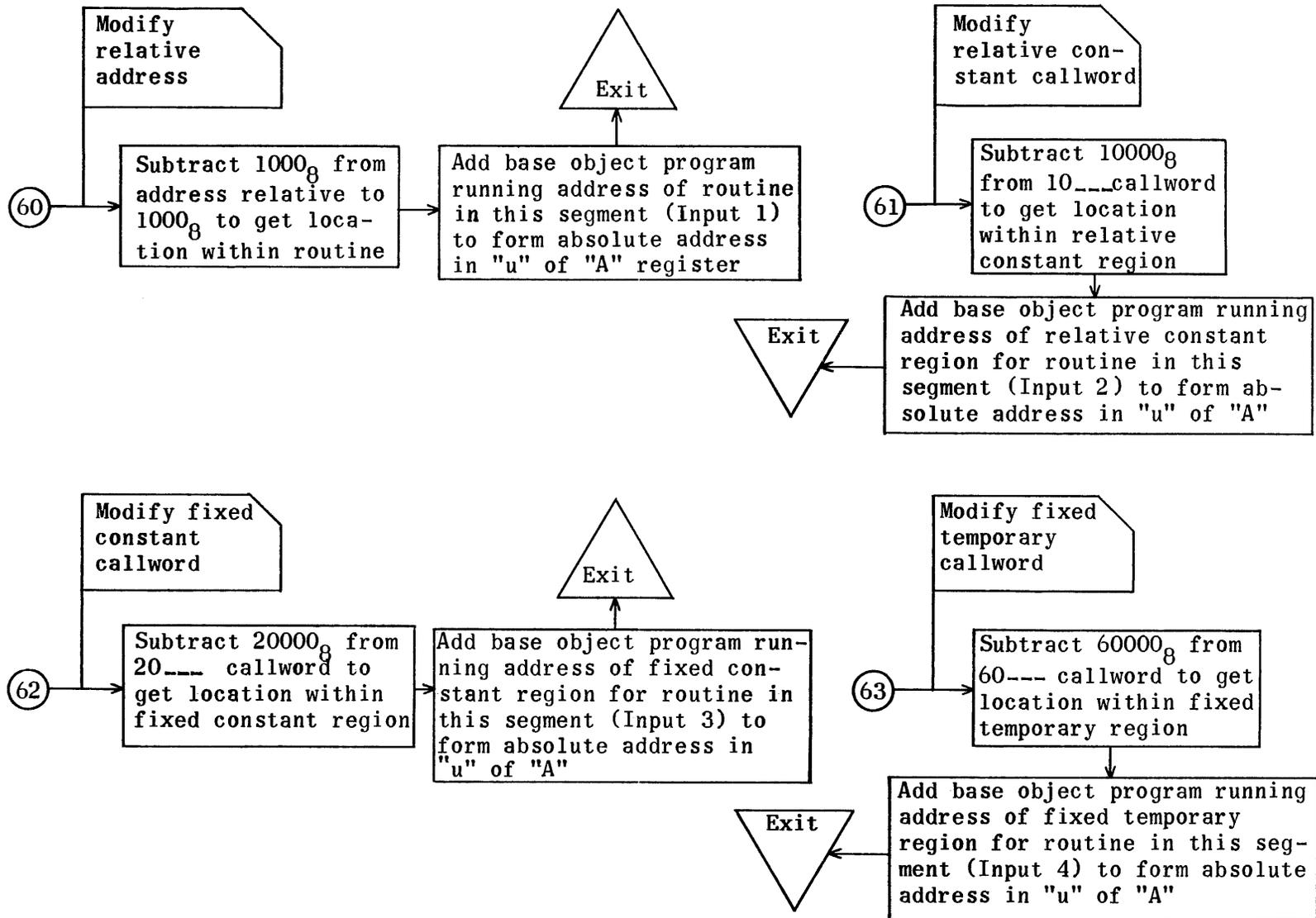


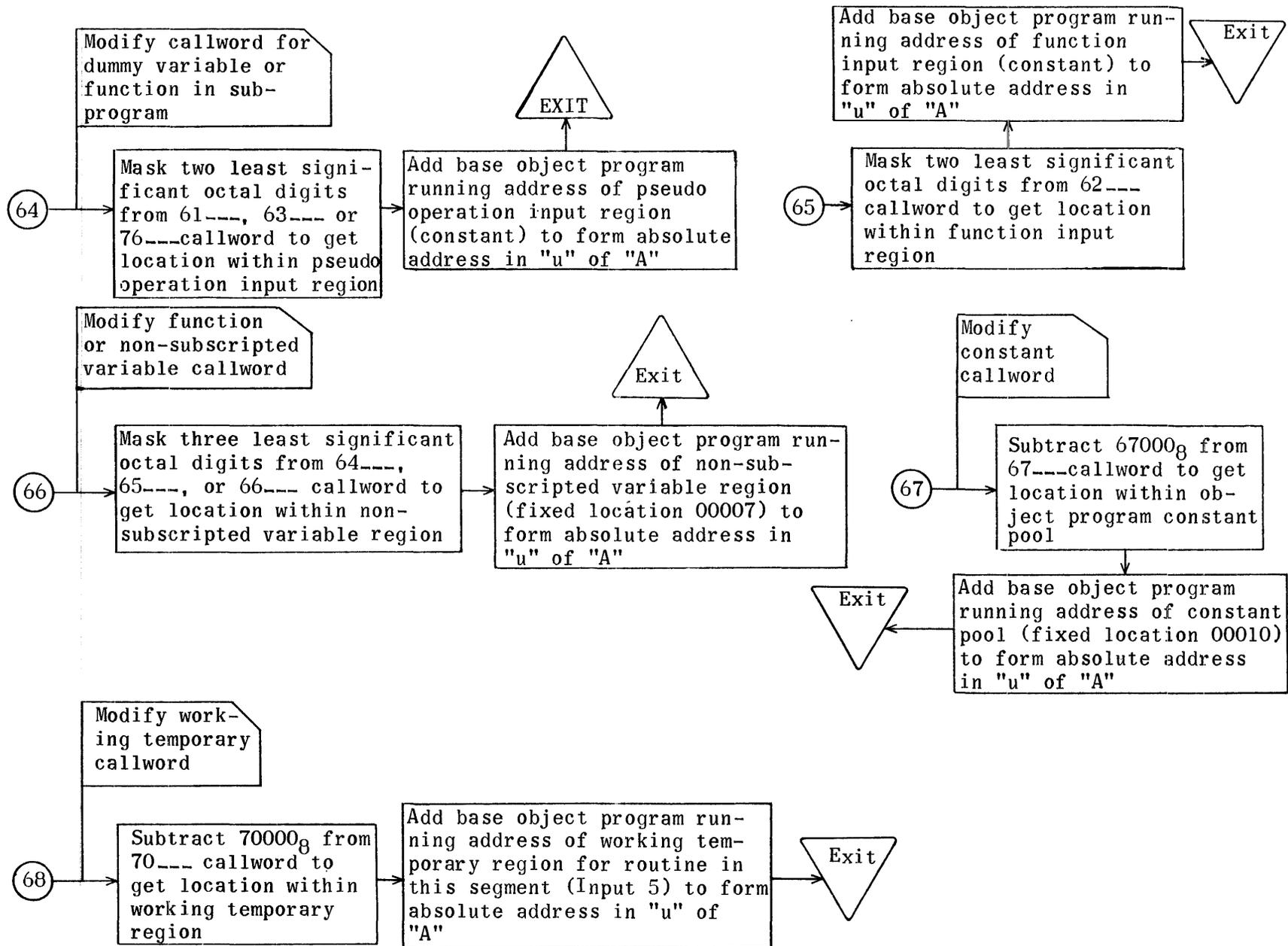
(MS) Modify Address Subroutine

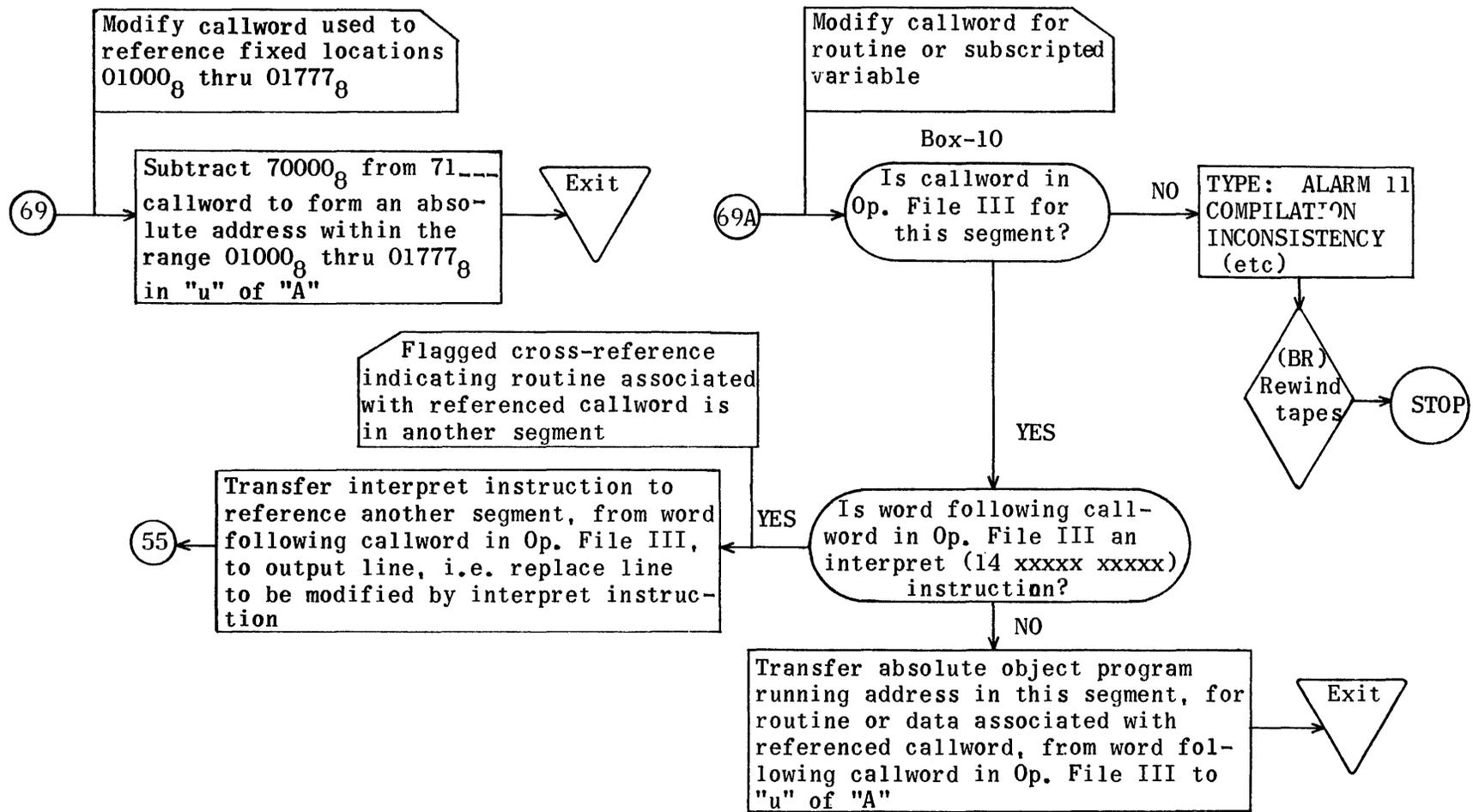
Input = Address to be modified in "u" of "Q" register
Output = Modified (absolute) address in "u" of "A" register



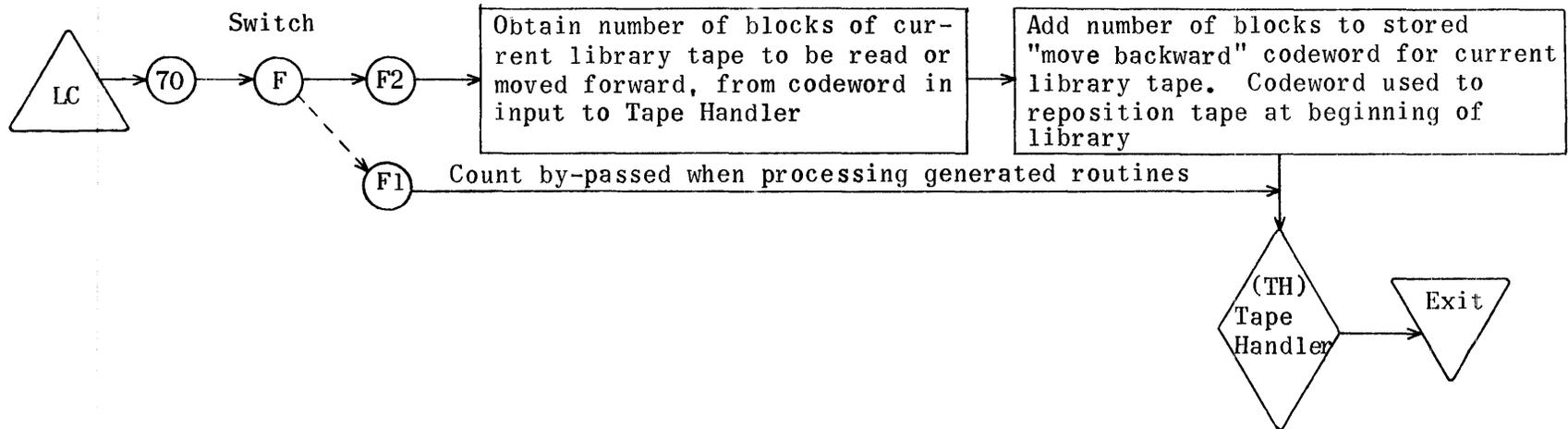




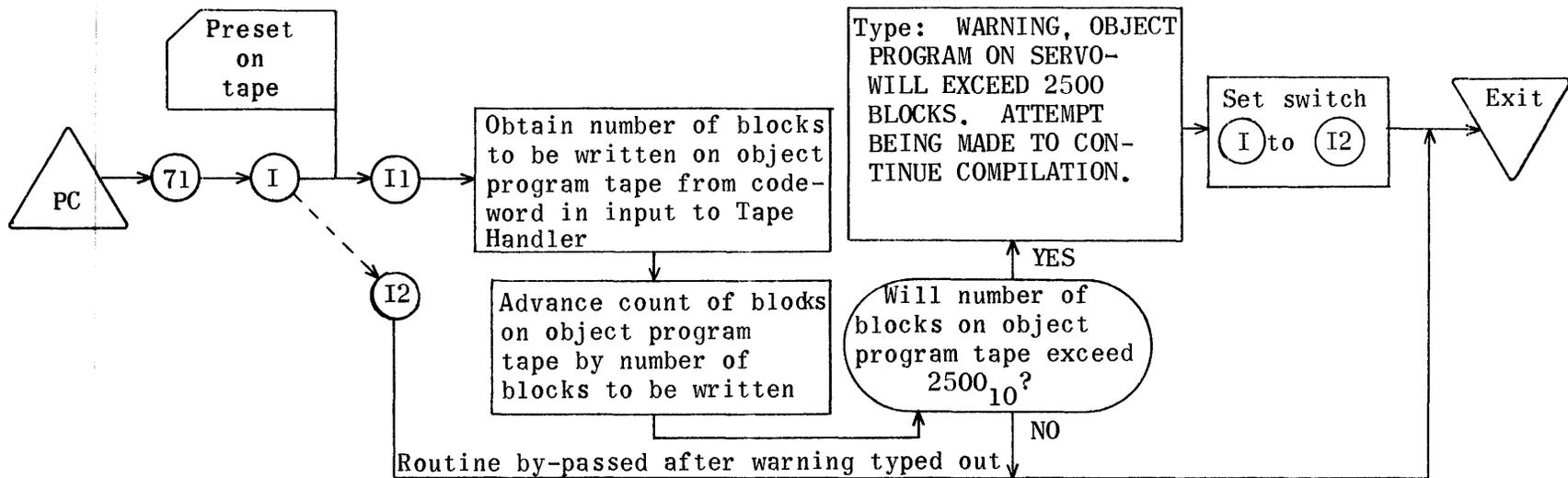




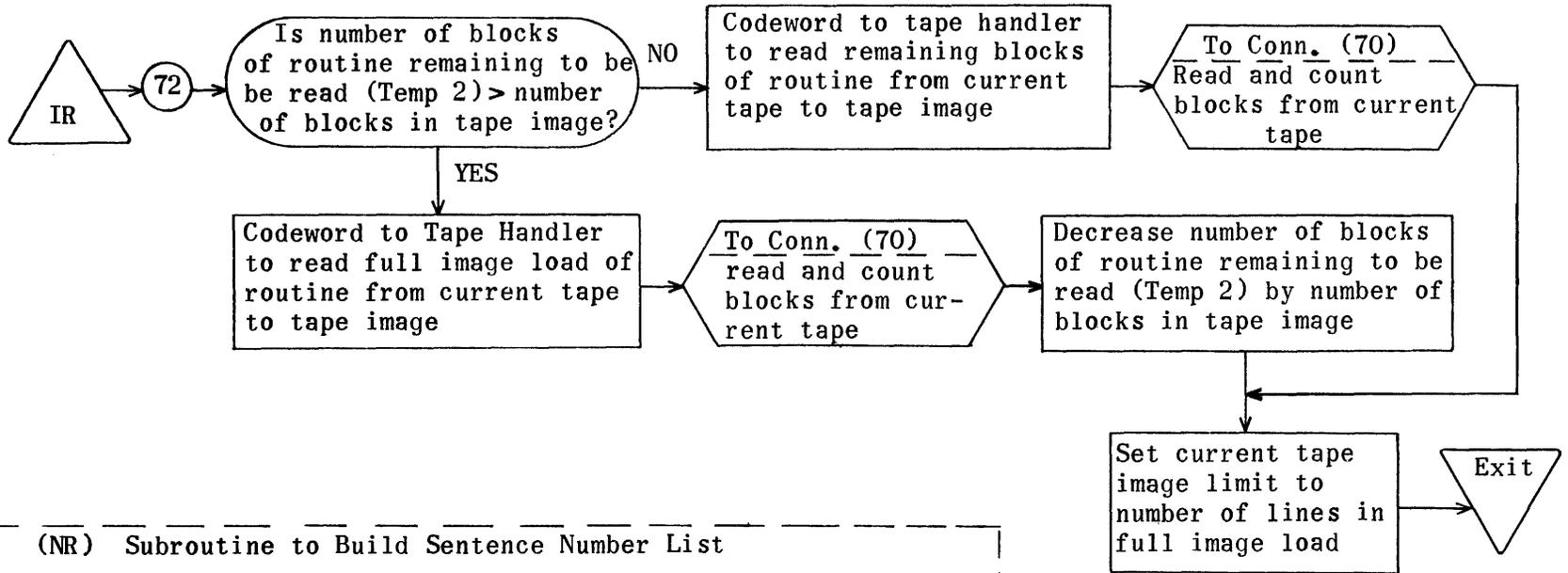
(LC) Subroutine to count blocks of Library (Fixed or Standard) Processed



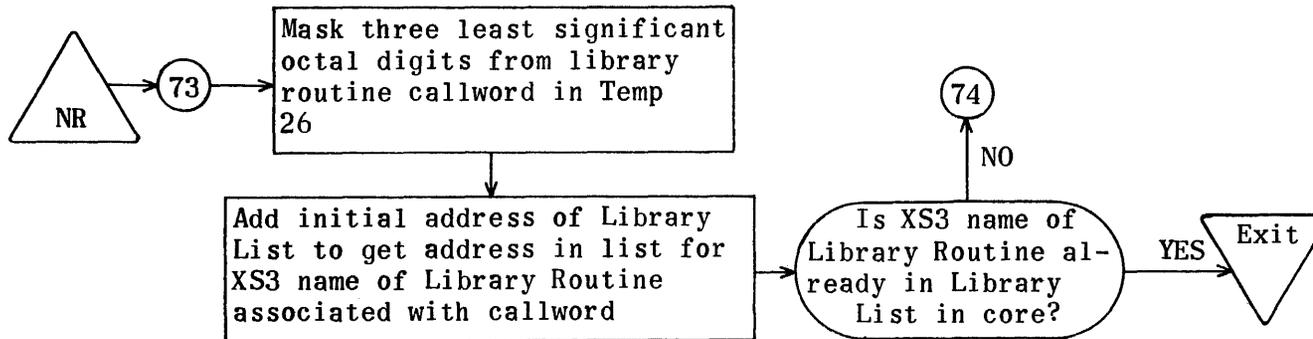
(PC) Subroutine to Count Blocks on Object Program Tape

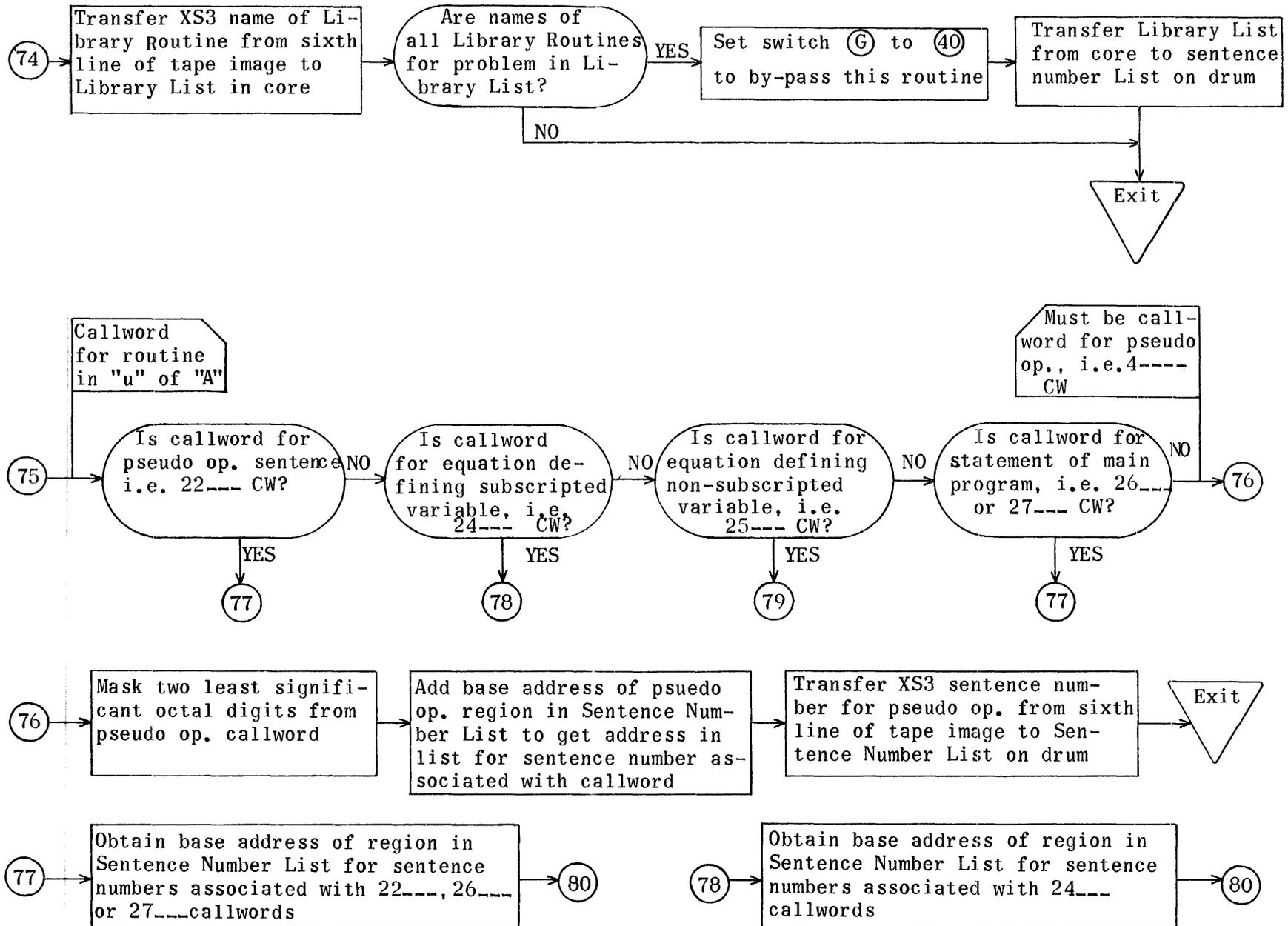


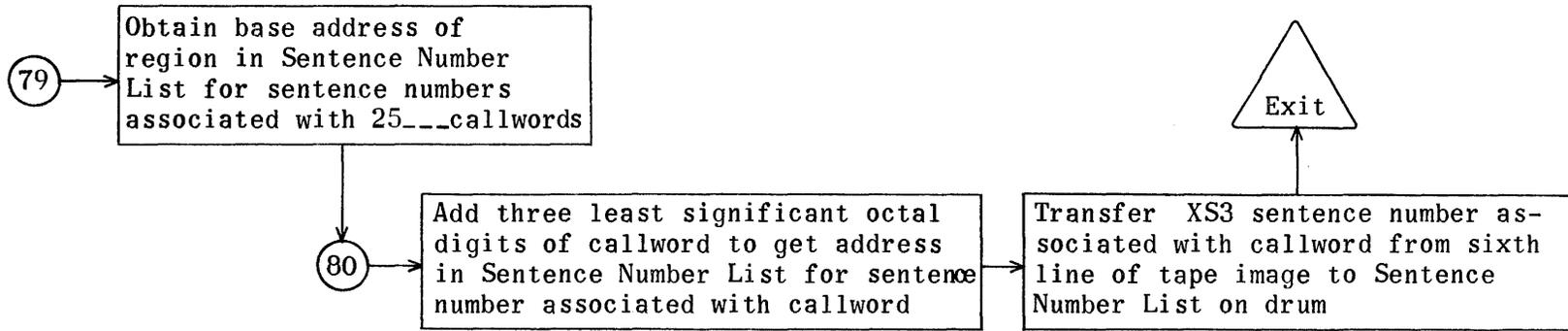
(IR) Input Routine



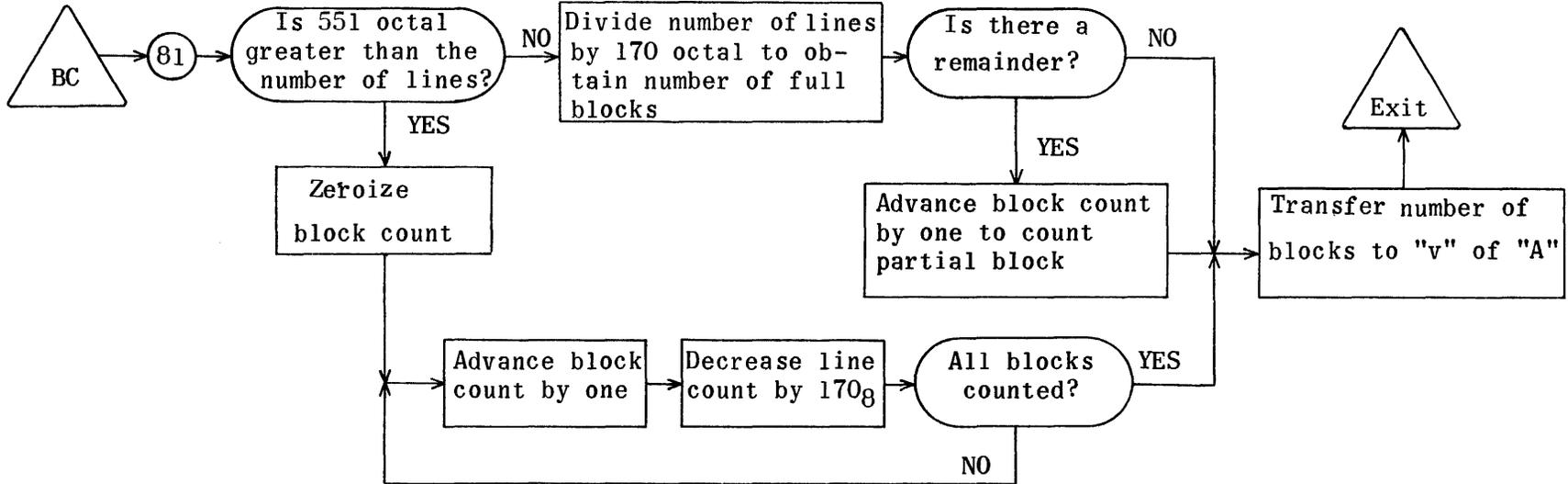
(NR) Subroutine to Build Sentence Number List







(BC) Subroutine to calculate block count from line count
 Input = Number of lines in "v" of "A"
 Output = Number of blocks in "v" of "A"



PROCESSOR REGIONS

RE	TN20	Servo Indicator
RE	TH21	Tape handler
RE	UP421	Uniprint
RE	BR537	
RE	CK653	
RE	CL701	
RE	CM715	
RE	CN740	
RE	CP753	
RE	CQ1004	
RE	CR1033	
RE	CS1065	
RE	CT1117	
RE	CU1134	
RE	CV1174	
RE	CW1226	
RE	CX1251	
RE	CY1310	
RE	CZ1350	
RE	DA1401	
RE	LC1424	
RE	PC1433	
RE	IR1444	
RE	NR1460	
RE	BC1521	
RE	FC1536	
RE	RC1570	
RE	TC1627	
RE	TL1660	
RE	TO1667	
RE	LV1712	
RE	MR1724	
RE	MS1751	
RE	MT2001	
RE	MU2040	
RE	MC2056	
RE	MD2105	
RE	MI2131	
RE	TS2140	

RE	LL2240	List of library routine names
* RE	TI3240	Tape image
* RE	FA4200	File image
RE	DD40101	Modified dimension list
RE	ND42102	Sentence number list
RE	DI46202	Segment image on drum
RE	IL740	Number of lines in tape image
RE	FL3600	Number of lines in file image
RE	BL4540	Number of lines in buffer, i.e., BL = IL + FL
RE	IB4	Number of blocks in tape image
RE	BB24	Number of blocks in buffer, i.e., tape image + file image
RE	PB4705	Limit number of blocks on object program tape
RE	II4200	Limit address for full image load, i.e., II = TI + IL
RE	FI610	
RE	PI624	
RE	ZA77000	Entrance to Unicode service routines
END		

* NOTE: The tape image and file image form the buffer for writing the segment on tape; hence, they must be adjacent in memory with the tape image appearing first.

Processor

	IA	CK		Begin Processor
0	MJ	0	ZA10	Exit → Unicode Service Routine
1	TP	5	Q	Library Indicator Word → Q
2	QT	FC5	A	# Library routines in problem → Av
3	ST	FC3	TS22	# Library routines in problem → "v" of temp.
4	QJ	CK6 yes	CK5 no	Fixed library required?
5	TV	RC27	CM22	Set switch (A) → (A2)
① 6	QJ	CK10 yes	CK7 no	Standard library required?
7	TV	RC31	CN12	Set switch (B) → (B2)
10	LA	LV6	6	# blks in buffer → tape codeword position
11	AT	TC26	TC26	Form codeword to write full buffer load on object program tape
12	LQ	LV7	6	# blks in tape image → codeword position
13	LQ	LV11	6	Limit # blks object program tape → codeword position
14	TP	TS22	TS14	# Library routines - 1 → library list index
15	TP	FC30	TS13	Preset count blks. binary tape → Max. # blks. initialization
16	TP	12	Q	Segment image address → Qv
17	QT	FC5	TS25	Segment image address → "v" of temp
20	RS	RC17	TS25	Drum add. for seg. - Run. add. seg. = Drum add. corres. to loc. zero
21	SP	TN	0	Servo layout indicator
22	ZJ	CK23	CL	(A) = zero ⇒ 5 servos; (A) ≠ zero ⇒ 7 servos
23	RA	T06	FC31	Set Object program servo # = 6 → printout
24	RP	20012	CL	
25	RA	TC17	TN	Setup tape codewords for 7 servo layout
	CA	CK26		

	IA	CL		
②	0	TP TC14	TH3	
	1	RJ TH2	TH	Read 1 blk. Op. File III (servo 5) → File image
	2	TP FA	A	1st word Op. File III image → A
	3	EJ TL1	CL6 yes	1st word = File Δ 3? Yes ⇒ entry label blk. for new segment
	4	EJ TL	DA	1st word = zzzzzz? Yes ⇒ End tape blk; End processing
	5	MJ 0	BR12	Alarm 10
③	6	RP 17777	CL10	Zeroize 277758 word segment image on drum to allow for possible 3 core bank running segment.
	7	TP FC	DI	
	10	RP 17777	CL12	
	11	TP FC	DI7777	
	12	RP 17777	CM	
	13	TP FC	DI17776	
		CA CL14		

	IA	CM		
	0	TV RC1	CS31	Set switch D → D1
	1	TV RC32	CV31	Set switch E → E1
	2	TV RC32	CW2	Set switch J → E1
	3	TV RC20	LC1	Set switch F → F1
	4	TP TS22	TS15	# Library routines in problem
	5	TP TC22	TS16	- 1 → Segment Index
	6	TP TC20	TS17	Set current tape codeword to read [n] blks. Gen. tape
	7	SP LV7	0	Set current tape codeword to read 1 blk. Gen. tape
	10	AT TC22	TS20	Set current tape codeword to read full image load Gen. tape
	11	TP TC17	TS21	Set current tape codeword to read move forward [n] blks. Gen. tape
④	12	RJ CR	CQ	Read Op. File III → image and process generated routines
	13	TP TC23	TH3	Rewind generated routine tape
	14	RJ TH2	TH	
	15	TV RC24	CS31	Set switch D → D2
	16	TV RC33	CV31	Set switch E → E2
	17	TV RC33	CW2	Set switch J → E2
	20	TV RC21	LC1	Set switch F → F2
	21	TV RC25	CR6	Set switch C → C2
Fixed Library Switch	22	MJ 0	[CN2]	
①		CA CM23		

	IA	CN		
⑤	0	TP TC2	TH3	} Move servo 1 backward to beginning of fixed library
	1	RJ TH2	TH	
	2	TP TC3	TC2	
	3	TP RC22	LC4	Reset move backward codeword → zero blks.
	4	TP TC5	TS16	Set to count blks fixed library processed
	5	TP TC4	TS17	Set current tape codeword to read [n] blks fixed library
	6	SP LV7	0	Set current tape codeword to read 1 blk fixed library
	7	AT TC5	TS20	Set current tape codeword to read full image load fixed library
	10	TP TC1	TS21	Set current tape codeword to move FW [n] blks
	11	RJ CR	CR27	Process fixed library routines
Switch } ⓑ	12	MJ 0 CA CN13	[CP2]	

	IA	CP			
⑥	0	TP	TS23	TH3	Move backward library tape (servo 2) to library routines entry label. Reset move backward codeword → zero blks Setup to count blks of library tape processed Set current codeword to read [N] blks library tape Set current codeword to read 1 blk. library tape Set current codeword to read full image load library tape Set current codeword to move [N] blks. library tape
	1	RJ	TH2	TH	
	2	TP	TC7	TS23	
	3	TP	RC23	LC4	
	4	TP	TC12	TS16	
	5	TP	TC10	TS17	
	6	SP	LV7	0	
	7	AT	TC12	TS20	
	10	TP	TC6	TS21	
	⑦	11	TP	TC11	
12		RJ	LC	LC1	
13		TP	TI	A	
14		EJ	TL4	CP16	
15		MJ	0	BR7	
⑧	16	RP	30170	CP20	Transfer 1st block library routines from 2nd blk → 1st block tape image Process STANDARD LIBRARY ROUTINES
	17	TP	TI170	TI	
	20	RJ	CR	CR1	
Switch	21	MJ	0	[CP22]	Fixed library required? Set switch ① → ① after 1st seg- ment
	22	TP	5	Q	
⑨	23	QJ	CP24 yes	CP25 no	Standard library required? Set switch ② → ② after 1st seg- ment
	24	TV	RC26	CM22	
	25	QJ	CP26 yes	CP27 no	By-Pass preceding setups after 1st segment
	26	TV	RC30	CN12	
	27	TV	CP30	CP21	
	30	MJ	0	CX	
		CA	CP31		

	IA	CQ			
②6	0	SP FA3	0		# words Op. File III → Au
	1	SA FC1	0		Add j=2 to # words Op. File III to form jn
	2	TU A	MU2		jn to search Op. File III → "RP" in Add. mod. rtn.
	3	TU A	CR14		jn to search Op. File III → "RP" in processor
	4	TP FA6	TS7		# lines in preface → "u" of Temp.
	5	TP FA2	TS10		Segment No. → Temp.
	6	TP FA4	TS12		# lines statements and routines + 2 → Temp.
②7	7	TP FA5	TS11		Address for 'IP' jump to next segment → Temp.
	10	SP FA3	25		# words Op. File III → "v" of A ₁
	11	LT 0	A		# words Op. File III → A _r
	12	DV FC2	Q		# words Op. File 3/170g = # full blks. Op. File III
	13	ZJ CQ14 yes	CQ15 no		Is there partial blk?
	14	RA Q	FC3		Adv. # blks. by 1 to count partial block
	15	SP Q	25		# blks. Op. File III → A in tape codeword position
	16	AT TC15	TH3		Codeword to read [N] blks servo 5 → tape handler
	17	RJ TH2	TH		Read Op. File III from servo 5 → Op. File III image
②8	20	TP TC21	TH3		Codeword to read 3 blks Gen. rtn. tape (4 or 7) → tape handler
	21	RJ TH2	TH		Read 2 label blks and 1st blk. gen. rtns → tape image
	22	TP TI170	A		1st word of 2nd label blk → A
	23	EJ TL2	CQ25 yes		Label = ΔSUBRO? (i.e. gen. rtn. tape positioned correctly?)
	24	MJ 0	BR11		Alarm 9
	25	RP 30170	CR1	}	Trans. 1st blk. gen. rtns. (incl. prelude,) from 3rd blk. → 1st blk. tape image
	26	TP TI360	TI		
		CA CQ27			

	IA	CR			
②9	0	MJ	0	[30000]	
	1	TP	TI	Q	1st word tape image → Q
	2	QT	MC13	A	Op. code → A
	3	ZJ	CR no	CR4 yes	Op. code = 0? No ⇒ end of fixed library, gen. rtns, or lib. rtns.
	4	QT	FC5	TS6	# lines prelude and routine → Temp.
	5	QT	MC	TS26	Routine callword → Au and Temp. 4
Switch } C	6	RJ	NR	[NR17]	Sentence number for routine → sentence number List
③0	7	TP	TS6	A	# lines prelude and routine → Av
	10	RJ	BC	BC1	# blks. prelude and routine → Av
	11	SS	FC3	25	# blks. prelude and routine-1 → Au in codeword position
	12	TP	A	TS2	# blks. prelude and routine-1 → Temp. 3 in codeword position
	13	SP	TS26	0	Routine callword → Au
	14	RP	[30000]	CR24	No ⇒ move past routine → 1st blk. next routine
	15	EJ	FA	CR16	Callword in Op. File III?
	16	SN	Q	17	-jn + r → Au
	17	SA	CR14	0	+r → Au
	20	SA	CR15	0	Address of word following callword in Op. File III → Au
	21	TU	A	CR22	Address of word following callword in Op. File III → NI
	22	TP	[30000]	A	Word following callword → A
	23	TJ	MC23	CS no	Word following callword "IP" inst? (i.e. 14-00000-00000 > (A)?)
③1	24	SP	TS2	0	# blks prel. and rtn-1 → Au in codeword position
	25	AT	TS21	TH3	Codeword → move forward [N] blks. routine (gen. or lib.) tape
	26	RJ	LC	LC1	Move past routine and count blks moved
③2	27	TP	TS17	TH3	Codeword to read 1 blk routine (prelude blk.) → G.T.H.
	30	RJ	LC	LC1	Read 1st block of next routine → tape image
	31	MJ	0	CR1	
		CA	CR32		

General Setup for Routine Modification

	IA	CS		
③③	0	TP A	Q	Word following callword in Op. File III → Q
	1	QT FC5	TS3	H.S.S. running add. rtn. to be modified → temp 3
	2	LT 10017	MI1	H.S.S. running add. rtn. to be modified → input mod. rtn.
	3	QA MC	MI6	H.S.S. running add. + # lines in rtn. → "u" of temp.
	4	TP TI2	Q	3rd word or prelude → Q
	5	QT FC4	TS	# fixed constants → temp. 0 in "v"
	6	LQ Q	6	# relative constants → Qu
	7	QT MC2	MI2	# relative constants → mod. rtn. input
	10	LQ Q	11	# fixed constants → Qu
	11	QT MC2	MI3	# fixed constants → mod. rtn. input
③④	12	LQ Q	11	# fixed temporaries → Qu
	13	QT MC2	MI4	# fixed temporaries → mod. rtn. input
	14	LQ Q	11	# working temporaries → Qu
	15	QT MC2	MI5	# working temporaries → mod. rtn. input
	16	SP MI6	0	Last address in running rtn. + 1 → Au
	17	ST MI5	MI5	Initial running add. working temps → input mod. rtn.
③⑤	20	ST MI4	MI4	Initial running add. fixed temps → input mod. rtn.
	21	ST MI3	MI3	Initial running add. fixed constants → input mod. rtn.
	22	ST MI2	MI2	Initial running add. relative constants → input mod. rtn.
③⑥	23	RA TS3	RC17	Form drum image address of modified routine → "v" of temp 3
	24	TP TI1	TS1	No. lines subject to add. mod. → temp 1 (counter 1)
	25	TP RC34	CU	Preset add. 1st line to be modified → 7th line tape image
	26	TP RC3	CU4	Preset add. for 1st modified line → 1st line tape image
③⑦	27	TV TS3	CU26	Preset drum image add. for rtn. to be modified
	30	TP LV	LV2	Set current image limit → 1 blk.
Setup Switch	31	MJ 0	[30000]	
①		CA CS32		

Special Setup For Library Routine Modification

③⑧	0	IA CT TP TI6	Q	Entrance line of library rtn. → Add. mod. rtn input
	1	RJ MR	MR1	Modify entrance line
	2	TP MI	TI6	Modified entrance line → routine heading
	3	RA TI4	TI3	# inputs + # outputs
	4	AT FC11	TI4	# lines in lib. rtn hdg. (3) + # inputs + # outputs → Av
	5	AT CU4	CU4	Adv. add. for next modified line by # inputs + # outputs + 3
	6	RS TS1	TI4	Decrease # lines subj. to add.mod. to exclude hdg. and inputs and outputs
	7	LA TI4	17	# inputs + # outputs + 3 → Au
	10	AT RC1	CT13	Setup jn of repeat to move hdg. → Beginning tape image
③⑨	11	RA CU	TI4	Adv. add of next line to be modified to skip hdg. + inputs + outputs
Switch } ⓐ	12	RJ NR	[NR1]	Library routine name → sent. no. List
④⑩	13	[O 30000	30000]	Move routine heading to beginning of tape image
	14	TP TI6	TI	
		CA CT15		

Process Lines Of Routine To Be Modified

④1	0	IA CU			
	1	TP [30000]	Q		Line to be modified → Q
		QT MC13	A		Op. code of line to be modified → A
	2	EJ FC12	CU7		Op. code = 10? ⇒ increment last modified line by (u and v)
	3	RJ MR	MR1		Modify line
	4	TP MI	[30000]		Modified line → tape image
	5	RA CU4	FC3		1 in "v" adv. → add. for next mod- ified line
④2	6	MJ 0	CU20		
	7	SP CU4	0		Address for next modified line → Av
	10	EJ RC3	CU13		Add. next mod. line = 1st line tape image? Yes ⇒ last mod. line on drum
	11	SS FC3	17		H.S.S. address last line modified → Au
	12	MJ 0	CU15		
	13	SP CU26	0		Address of line following last mod- ified line on drum → Av
	14	SS FC3	17		Drum address of last line modified → Au
	15	TU A	CU17		
	16	RS Q	FC12		"u" and "v" of "10" line → Q with zero Op. code
	17	RA [30000]	Q		Add contents of "u" and "v" of "10" line → last modified line
④3	20	RA CU	FC13		1 in "u" adv. → address of next line to be modified
④4	21	TJ LV2	CU33 no		Was this last line in current image load?
	22	RS CU4	RC3		Determine # modified lines → Av
	23	SA FC14	17		Form jn of repeat to transfer mod- ified lines → drum image
	24	TU A	CU25		Preset jn of repeat
	25	RP [30000]	CU27	}	
	26	TP TI	[30000]		Transfer modified lines → drum image
	27	RA CU26	CU4		Adv. add. in drum image by # lines transferred
	30	RJ IR	IR1		Read more lines of gen. routine → tape image
	31	TP RC2	CU		Preset address next line to be modified → 1st line tape image
	32	TP RC3	CU4		Preset address for next modified line → 1st line tape image
④5	33	IJ TS1	CU		Have all lines subject to address modification been processed?
	34	SP CU4	0		Address for next modified line in tape image → Av

35	ST	RC3	TS4	# modified lines not yet transferred to drum → temp
36	SP	TS	0	Number of fixed (unmodifiable) constants → Av
37	ZJ	CV no	CW yes	Number fixed constants = zero?
	CA	CU40		

	IA	CV		
④6	0	SP LV2	0	Limit for current image load → A
	1	SS CU	25	Form # lines in image not processed → "v" of A ₁
	2	LT 0	A	Form # lines in image not processed → Av
	3	TJ TS	CV5 yes	# fixed constants > # lines in image not processed?
	4	SP TS	0	# fixed constants (all const. for rtn) → Au
	5	TP A	TS5	# fixed constants in image to be transferred to drum → temp. 5
	6	SP TS4	0	# modified lines yet to be trans- ferred to drum → Au
	7	ZJ CV10 no	CV20 yes	# modified lines yet to be trans- ferred to drum = Zero?
	10	TU CU	CV17	Preset address for 1st fixed constant
	11	TV CU4	CV17	Preset transfer add. for fixed const. → Add. for next mod. line
	12	SP TS5	17	} Preset jn to pack fixed constants with modified lines
	13	AT RC4	CV16	
	14	RA TS4	TS5	# modified lines + # fixed const. = # lines to be trans. to drum → temp.
	15	TU RC2	CV26	Preset Add of 1st line to be trans. to drum → 1st line tape image
	16	[RP [30000]	CV22]	} Pack fixed constants with modified lines in tape image
	17	TP [30000]	[30000]	
	20	TP TS5	TS4	Set # lines to be trans. to drum = # fixed const. in tape image
21	TU CU	CV26	Preset add. of 1st line to be trans. to drum → add. 1st fixed const.	
22	SP TS4	17	} Preset jn to transfer fixed const. and/or modified lines to drum	
23	AT RC5	CV25		
④7	24	TV CU26	CV26	Preset next available address in drum image
	25	[RP [30000]	CV27]	} Transfer fixed constants and/or mod- ified lines → drum image
	26	TP [30000]	[30000]	
27	RA CU26	TS4	Advance drum image address by # lines transferred	
④8	30	RS TS	TS5	Decrease # fixed const. to be trans. by # just transferred
	Switch } ④E	31	ZJ CW3 no	[30000] yes
		CA CV32		

	IA	CW		
	0	TP TS4	TS	# modified lines yet to be transferred → TS
Switch } ④	1	SP TS4	17	# Modified lines yet to be transferred = zero?
	2	ZJ CW7 no	[30000] yes	
④	3	RJ IR	IR1	Read more fixed constants from gen. tape → tape image
	4	SP LV3	0	# lines full image load → Au
	5	TJ TS	CW14 yes	# fixed constants yet to be trans. > full image load?
	6	SP TS	17	# fixed constants yet to be trans. → Au
⑤	7	AT RC6	CW12	Preset jn of repeat
	10	TV CU26	CW13	Preset available drum image address
	11	RA CU26	TS	Adv. drum image address by # lines to be transferred
	12	[RP [30000]	CR27] }	Transfer remaining modified lines or fixed constants → drum image
	13	TP TI	[30000] }	
	14	TV CU26	CW20	Preset avail. drum image address
	15	RA CU26	LV3	Adv. drum image add. by # lines full image load
	16	RS TS	LV3	Reduce # fixed constants by # lines full image load
	17	RP IL30000	CW3	Transfer full image load fixed constants → drum image
	20	TP TI	[30000] }	
⑤	21	IJ TS15	CR27 no	All library rtns for problem processed this segment?
	22	MJ 0	CX	→ End segment
		CA CW23		

		IA	CX				
⑨	0	RA	TS11	RC17	Initial add. drum image + add. of "IP" jump to next seg. → temp. 7		
	1	TV	A	CX3	Preset drum image address for "IP" instruction		
	2	TP	FA1	A	"IP" Jump to next segment from 2nd word Op. File III → "A"		
	3	AT	FC3	[30000]	Add one to "v" address and transfer "IP" to drum image		
	4	TP	TL5	TI	SEGMEN → 1st word in segment label block		
	5	TP	TL6	TI1	T△△△△△ → 2nd word in segment label block		
⑩	6	SP	TS10	25	Octal segment no. → "v" of "A" left		
	7	LT	0	TI2	Octal seg. no. → "v" of 3rd word in seg. label block		
	10	SP	TS7	71	# lines in Preface → "v" of "A" right		
	11	TP	A	TI7	# lines in Preface → "v" of 8th line seg. lab. blk.		
	12	DV	FC2	TI3	# lines in Pref. (term)/170g = full blks term. → 4th line seg. lab. blk.		
	13	ZJ	CX14	yes	CX15	no	Is there partial block?
⑪	14	RA	TI3	FC3	Adv. # blks. Termination by one to count partial blk.		
⑫	15	SP	TI7	0	# lines in preface → Av		
	16	AT	TS12	TS1	# lines in Preface + # lines stmts and routines + 2 → temp 2		
	17	DV	FC2	TI4	# full blocks segment+Preface		
⑬	20	LQ	TI4	25	→ 2nd thru 5th octal digit positions of 5th line seg. lab. blk.		
	21	TP	A	TI5	# lines partial blk. seg. + Pref. → "v" of 6th line seg. lab. blk.		
	22	TP	TS25	A	Seg. image address → A		
	23	AT	TS1	TS5	Seg. image add. + # lines seg. + Pref. = add. line after Preface		
⑭	24	SS	TI7	17	(A) - # lines Pref. = Pref. Exit add. → Au		
	25	SA	TS5	0	Add. line following Pref. → Av		
	26	SS	FC20	0	(A) - 2 = Pref. Ent. add. → Av		
	27	AT	FC17	TI6	"RJ" inst. to execute Preface → 7th line seg. lab. blk.		
⑮	30	SP	TI5	17	# lines partial blk. seg. + Preface → Au		
	31	ZJ	CX32	no	CX34	yes	# lines partial blk. seg. + Preface = zero?

①6

32	SA	TS5	0	Address of line following preface → Av
33	ST	TI5	TI5	# lines partial blk. seg. + pref. in "u" and H.S.S. part. blk. in "v" → 6th line seg. lab.
34	TP	TI4	TS2	# full blks. seg. + pref. → temp. in codeword position
35	RP	10160	CY	
36	TP	TL	TI10	Z's → lines 9-120 of segment label blk.
	CA	CX37		

	IA	CY			
	0	SP	TI3	25	# blks. in pref. (term.) → A in codeword position
	1	AT	TC15	TS	Codeword to read Termination from tape → Temp.
	2	AT	FC7	TH3	Adv. by 1 to include Op. File III end entry blk.
	3	RJ	TH2	TH	Read Preface and/or Op. File III end entry blk. → File image
	4	TP	FA	A	1st word File image → A
	5	EJ	TL3	CY7	1st word = END△OF? (i.e. tape positioned properly)
①⑦	6	MJ	0	BR12	Alarm # 10
	7	RA	TS7	FC15	Add 30000 to # lines in Preface → Au
	10	TU	A	CY13	Preset "u" of repeat
	11	RA	TS12	RC35	# lines stmts. and rtns + 2 + drum add. init. stmt. = drum add. Preface
	12	TV	A	CY14	Preset drum add. for Preface
	13	RP	[30000]	CY15	} Preface → Drum image
	14	TP	FA170	[30000]	
	15	RP	30170	CY17	
	16	TP	TI	DI	
①⑧	17	RA	TS1	FC2	Segment label block → drum image
	20	TU	RC11	CY26	Adv. # lines segment by 170 ₈ to count label blk.
①⑨	21	SP	LV5	0	# lines in buffer (tape image + file image) → Av
	22	TJ	TS1	CZ17	# lines segment remaining > full buffer load?
②①	23	SP	TS1	17	} Form jn of repeat → NI
	24	AT	RC7	CY25	
	25	[RP	30000	CY27]	
	26	TP	[30000]	TI	
	27	TP	MC	Q	
	30	QT	DI5	Q	Segment + seg. label blk. → buffer "u" mask → Q
	31	ZJ	CY32 no	CZ1	# lines partial blk. seg. + Pref. → Qu and Au
	32	TP	RC10	A	# lines partial blk. seg. + Pref. = zero?
	33	ST	Q	CY36	} Form "RP" to fill rest of partial blk. with Z's
	34	TV	RC3	CY37	
	35	RA	CY37	TS1	Preset "v" of "TP" to initial add. tape image
	36	RP	[30000]	CZ	Adv. by # lines segment + label → Add. for Z's in partial blk.
	37	TP	TL	[30000]	Z's → fill remainder partial block
		CA	CY40		

	IA	CZ		
0	RA	TS2	FC7	Adv. # full blks. seg. + Pref. by 1 to count partial block
1	RA	TS2	FC7	Adv. # full blks. seg. + Pref. by 1 to count seg. label block
2	AT	TC24	TH3	Codeword → tape handler
3	RJ	PC	PC1	Adv. and check count of blks. on Binary program tape
4	RJ	TH2	TH	Write remainder (or all) of segment on Binary program tape
5	TP	TS	Q	
6	TP	Q	TH3	Codeword → tape
7	QT	FC21	TS	# blks. Termination → A and temp. in codeword position
10	ZJ	CZ11 no	CZ16 yes	# blks. Termination = zero?
11	RJ	PC	PC1	Adv. and check count of blks. on Binary program tape
12	RJ	TH2	TH	Read Termination from servo 5 → File image
13	SP	TS	0	# blks. Termination → A
14	AT	TC25	TH3	Codeword → Tape Handler
15	RJ	TH2	TH	Write Termination on Binary Program Tape
16	MJ	0	CL	→ Process next segment
17	TU	CY26	CZ21	Preset drum image add. of next buffer load
20	RP	BL30000	CZ22	Full buffer load of segment from drum image → buffer
21	TP	[30000]	TI	
22	RA	CY26	LV10	Adv. drum image add. by # lines full buffer load
23	TP	TC26	TH3	Codeword to write full buffer load → Tape Handler
24	RJ	PC	PC1	Adv. and check count of blks. on Binary program tape
25	RJ	TH2	TH	Write full buffer load on Binary program tape
26	RS	TS1	LV5	Decrease # lines segment remaining by # lines full buffer
27	RS	TS2	LV6	Decrease # full blocks seg. + Pref. by # blocks in buffer
30	MJ	0	CY21	
	CA	CZ31		

	IA	DA			
②③	0	TP	TC13	TH3	
	1	RJ	TH2	TH	Rewind library tape (servo #2)
	2	TP	TC16	TH3	
	3	RJ	TH2	TH	Rewind corrected problem tape (servo #5)
	4	RP	10360	DA6	} 2 blks. of Z's → buffer
	5	TP	TL	TI	
	6	TP	TC27	TH3	
	7	RJ	TH2	TH	Write 2 blks. of Z's on Binary program tape
②④	10	TP	TC30	TH3	
	11	RJ	TH2	TH	Rewind Binary program tape (servo # 3 or 6)
	12	RS	TC2	TC3	# blks. fixed library advanced
	13	SS	TC	0	# blks. adv. - total # blks. fixed library
	14	ZJ	DA15 no	CK yes	(A) = Zero? Yes ⇒ servo 1 positioned at listing phase setup blk.
②⑤	15	SJ	DA16 -	DA20 +	
	16	TN	A	A	# blks. to move forward → A
	17	TP	TC1	TC3	Replace move backward codeword by move fwd. CW
	20	AT	TC3	TH3	Move forward or backward codeword → G.T.H.
	21	RJ	TH2	TH	Position servo 1 at beginning listing phase setup block
	22	MJ	0	CK	
		CA	DA23		

Subroutine to Count Blocks of Library (Fixed or Standard) Processed

	IA	LC		
(70) Switch }	0	MJ 0	30000	
	1	MJ 0	[30000]	
(F)	2	TP TH3	Q	Tape handler codeword → Q
	3	QT FC21	A	# blks. current routine to be read or moved
	4	[0 30000	30000]	Add # blks. to move backward code- word for current tape
	5	RJ TH2	TH	→ tape handler
	6	MJ 0	LC	
		CA LC7		

Subroutine to Count Blocks on Object (Binary) Program Tape

	IA	PC		
	0	MJ	0	[30000]
⑦	} Switch	1	TP TH3	Q
①				Tape handler codeword → Q
	2	QT	FC16	A
				# blks. to be written on Binary prog. tape → A
	3	AT	TS13	TS13
				Adv. count of blks. Binary prog. tape by # blks. to be written
	4	TJ	LV11	PC
				Limit # blks. > current # blks. Binary program tape
	5	TP	TO	UP3
				Parameter → Uniprint
	6	RJ	UP2	UP
				Print warning
	7	TP	PC10	PC1
10		MJ	0	PC
		CA	PC11	

Input Routine

	IA	IR		
⑦	0	MJ 0	[30000]	
	1	SP LV7	0	
	2	TJ TS2	IR7yes	# blks. prelude and routine remain- ing to be read > # blks. tape image?
	3	SP TS2	0	
	4	AT TS16	TH3	Codeword to read [n] blks. current routine tape → tape handler
	5	RJ LC	LC1	Read remaining blks. current routine → image
	6	MJ 0	IR12	
	7	TP TS20	TH3	Codeword to read full image load current routine → tape image
	10	RJ LC	LC1	Read full image load current routine → tape image
	11	RS TS2	LV7	
	12	TP LV1	LV2	Set current image limit → # lines full image load
	13	MJ 0	IR	
		CA IR14		

Subroutine to Build Sentence Number List

	IA	NR		
⑦3	0	MJ	0	[30000]
	1	SP	TS26	0
	2	LT	22	Q
	3	QT	FC4	A
	4	AT	RC12	Q
	5	SP	A	17
	6	TU	A	NR10
	7	TP	TI5	A
	10	EJ	[30000]	NR yes
⑦4	11	TV	Q	NR12
	12	TP	A	[30000]
	13	IJ	TS14	NR no
	14	TV	RC36	CT12
	15	RP	31000	NR
	16	TP	LL	ND3100 }
⑦5	17	TJ	FC24	NR30 yes
	20	TJ	FC25	NR32 yes
	21	TJ	FC26	NR34 yes
	22	TJ	FC27	NR30 yes
⑦6	23	LT	17	Q
	24	QT	FC22	A
	25	AT	RC16	NR26 }
	26	[00	30000	30000]
	27	MJ	0	NR
⑦7	30	TP	RC13	A
	31	MJ	0	NR35
⑦8	32	TP	RC14	A
	33	MJ	0	NR35
⑦9	34	TP	RC15	A
⑧0	35	LQ	Q	25
	36	QA	FC4	NR37
	37	[0	30000	30000]
	40	MJ	0	NR
		CA	NR41	

Library routine callword → A

Add. for lib. rtn. name in lib.
list in core → Qv
Add. for lib. rtn. name in lib.
list in core → Au

XS3 library rtn name → A
Name in list?

XS3 library name → library list
in core

Names of all library routines for
problem in list?

Set switch ⑥ → ⑥2 to by-pass sent.
no. routine

Library list in core → sentence
no. list on drum

23000 > CW? Yes ⇒ pseudo Op.
sentence

25000 > CW? Yes ⇒ subs. var.
equation (NB → no routines with
23---- CW)

26000 > CW? Yes ⇒ non-subs. var.
equation

30000 > CW? Yes ⇒ statement of
main program

No ⇒ Pseudo Op.

Form add. for pseudo op. sent. no.
XS3 sent. no. for pseudo Op
→ sent. no. list on drum

Base add. for 22----, 26----, or
27---- CW

Base add. for 24---- CW

Base add. for 25---- CW

CW → Qv

Form add. for sent. no.

XS3 sent. no. → list

Routine to Calculate Block Count from Line Count

	IA	BC			Input = number of lines in "v" of "A"; Output = # blks. in Av	
⑧	0	MJ	0	[30000]		
	1	TJ	FC6	BC6	yes	5518 > # lines to be converted? (i.e. # blks. < 4?)
	2	DV	FC2	Q		# lines /170 ₈ = # full blks. → Q
	3	ZJ	BC4	BC13	yes no	Is there partial block?
	4	RA	Q	FC3		Adv. count of # blks. by 1 to count partial blk.
	5	MJ	0	BC		
	6	TP	FC	Q		Zero → Q
	7	TN	A	TS24		Complement of # lines to be converted → temp.
	10	RA	Q	FC3		Adv. count of blks. by 1 in "v"
	11	RA	TS24	FC2		Adv. complement of # of lines by 170 ₈
	12	SJ	BC10	BC13	yes	All blks. counted?
	13	TP	Q	A		# full blks. → Av
	14	MJ	0	BC		
		CA	BC15			

Fixed Constants

	IA	FC	
0	0	0	0
1	0	20000	0
2	0	0	170
3	0	0	1
4	0	0	777
5	0	0	77777
6	0	0	551
7	0	100	0
10	0	0	7
11	0	0	3
12	10	0	0
13	0	1	0
14	0	0	30000
15	0	30000	0
16	0	7700	0
17	37	0	0
20	0	0	2
21	07	77700	0
22	0	0	77
23	0	50000	0
24	0	23000	0
25	0	25000	0
26	0	26000	0
27	0	30000	0
30	0	3000	0
31	0	0	300
	CA	FC32	

Relative Constants

	IA	RC		
0	0	0	DI	
1	RP	30000	CU33	D1 in "v"
2	TP	TI	Q	
3	TP	MI	TI	
4	RP	30000	CV22	
5	RP	30000	CV27	
6	RP	30000	CR27	
7	RP	30000	CY27	
10	RP	10170	CZ	
11	0	DI	0	
12	0	0	LL	Initial address lib. name list in core
13	TP	TI5	ND	
14	TP	TI5	ND1000	
15	TP	TI5	ND2000	
16	TP	TI5	ND3000	
17	0	0	[DI170]	Chged. by program to drum add. for segment — H.S.S. add. for run. seg.
20	0	0	LC5	
21	0	0	LC2	
22	AT	TC2	TC2	
23	AT	TS23	TS23	
24	0	0	CT	D2 in "v"
25	0	0	CR7	
26	0	0	CN	A1
27	0	0	CN12	A2
30	0	0	CP	B1
31	0	0	CP21	B2
32	0	0	CR27	E1
33	0	0	CW21	E2
34	TP	TI6	Q	
35	0	0	DI170	Segment image address for first statement of segment
36	0	0	CT13	
	CA	RC37		

Tape Handler Codewords

	IA	TC		
0	0[0	017]00	0	# blocks of fixed library
1	3[0	000]01	0	Move forward [n] blks servo 1
2	[4[0	000]01	0]	Move backward codeword with count of fixed library advanced
3	4[0	000]01	0	Move backward [n] blks servo 1
4	5[0	001]01	TI	Read forward 1 blk servo 1
5	5[0	000]01	TI	Read forward n blks servo 1
6	3[0	000]02	0	Move forward n blks servo 2
7	4[0	000]02	0	Move backward n blks servo 2
10	5[0	001]02	TI	Read forward 1 blk servo 2
11	5[0	002]02	TI	Read forward 2 blks servo 2
12	5[0	000]02	TI	Read forward n blks servo 2
13	1 0	000 02	0	Rewind servo 2
14	5[0	001]05	FA	Read forward 1 blk servo 5
15	5[0	000]05	FA	Read forward n blks servo 5
16	1 0	000 05	0	Rewind servo 5
17	3[0	000]04	0	Move forward [n] blks servo 4 or 7
20	5[0	001]04	TI	Read forward 1 blk servo 4 or 7
21	5[0	003]04	TI	Read forward 3 blks servo 4 or 7
22	5[0	000]04	TI	Read forward n blks servo 4 or 7
23	1 0	000 04	0	Rewind servo 4 or 7
24	7 1	000]03	TI	Write [n] blks servo 3 or 6
25	7 1	000]03	FA	Write [n] blks servo 3 or 6
26	7 1	000]03	TI	Write full buffer servo 3 or 6
27	7 1	002]03	TI	Write 2 blks servo 3 or 6
30	1 0	000 03	0	Rewind servo 3 or 6
	CA	TC31		

Tape Labels

	IA	TL							
0	74	74747	47474	Z	Z	Z	Z	Z	Z
1	31	34463	00106	F	I	L	E	△	3
2	01	65672	55451	△	S	U	B	R	O
3	30	50270	15131	E	N	D	△	O	F
4	01	01463	42501	△	△	L	I	B	△
5	65	30324	73050	S	E	G	M	E	N
6	66	01010	10101	T	△	△	△	△	△
	CA	TL7							

Typeout

	IA	T0							
0	0	T01	22						
1	71	24545	03450	W	A	R	N	I	N
2	32	21010	15125	G	,	△	△	O	B
3	44	30266	60152	J	E	C	T	△	P
4	54	51325	42447	R	O	G	R	A	M
5	01	51500	16530	△	O	N	△	S	E
6	54	70510	10601	R	V	O	△	3	△
7	71	34464	60130	W	I	L	L	△	E
10	72	26303	02701	X	C	E	E	D	△
11	05	10030	30125	2	5	O	O	△	B
12	46	51264	56522	L	O	C	K	S	.
13	01	01246	66630	△	△	A	T	T	E
14	47	52660	12530	M	P	T	△	B	E
15	34	50320	14724	I	N	G	△	M	A
16	27	30016	65101	D	E	△	T	O	△
17	26	51506	63450	C	O	N	T	I	N
20	67	30012	65147	U	E	△	C	O	M
21	52	34462	46634	P	I	L	A	T	I
22	51	50227	77777	O	N	.	77	77	77
	CA	T023							

Limiting Values

	IA	LV		
0	TP	TI170	Q	Limit for 1 blk image load
1	TP	II	Q	Limit for full image load II = TI + IL
2	TP	30000	Q	Limit for current image load
3	0	0	IL	
4	0	0	FL	
5	0	0	BL	
6	0	BB	0	Shift to codeword position - # blks in buffer
7	0	IB	0	Shift to codeword position - # blks tape image
10	0	BL	0	# lines in buffer
11	0	PB	0	Limit # blks. Binary prog. tape (250 ₁₀) in codeword position
	CA	LV12		

Address Modification Control Subroutine

	IA	MR		Input = word to be modified in Q
⑤⑤	0	MJ	0	30000
⑤②	1	TP	Q	MI
	2	QT	MC13	MI6
	3	TP	MI6	A
	4	RP	20011	MR6
	5	EJ	MC3	MR17
	6	EJ	MC22	MR
	7	EJ	MC23	MR
	10	EJ	MC24	MR
	11	TP	MI	Q
	12	RJ	MS	MS1
	13	TU	A	MI
⑤③	14	TP	MI6	A
	15	RP	20006	MR17
	16	EJ	MC14	MR
⑤④	17	SP	MI	17
	20	TP	A	Q
	21	RJ	MS	MS1
	22	LT	25	Q
	23	TV	Q	MI
	24	MJ	0	MR
		CA	MR25	

Input = word to be modified in Q

Word to be modified → Output line

Oper. code of word to be modified
→ temp.

Oper. code of word to be modified
→ A

Op. code = 17, 22, 45, 56, 61, 63,
75, 76, or 77 ⇒ only "v" address

to be modified

Oper. code = 05? }

Oper. code = 14? }

Oper. code = 57? } Neither address

to be modified
"u" of word to be modified → "u"
of Q

Modify "u" address (result in
"u" of A)

Modified "u" address → output line

Oper. code of word to be modified
→ A

Op. code = 31, 32, 33, 34, 54, or
55 ⇒ only "u" address to be mod-
ified.

"v" of word to be modified → "u"
of A

"v" of word to be modified → "u"
of Q

Modify "v" address (result in "u"
of A)

Modified "v" add. → "v" of Q

Modified "v" add. → "v" of output

Modify Address Subroutine

	IA	MS		
⑤6	0	MJ	0	30000
	1	QT	MC	A
	2	RP	30024	MU2
	3	TJ	MD	MS4
	4	MJ	0	MS
	5	MJ	0	MT
	6	MJ	0	MT3
⑤7	7	MJ	0	MS
	10	MJ	0	MT6
	11	MJ	0	MS
	12	MJ	0	MU2
	13	MJ	0	MS
	14	MJ	0	MU2
	15	MJ	0	MT11
⑤8	16	MJ	0	MT14
	17	MJ	0	MT20
	20	MJ	0	MT14
	21	MJ	0	MT24
	22	MJ	0	MT30
	23	MJ	0	MT34
	⑤9	24	MJ	0
25		MJ	0	MS
26		MJ	0	MT20
27		MJ	0	MT14
	CA	MS30		

Address to be modified → Au with "op." and "v" = zero
 No ⇒ 77--- CW.
 Search list to determine modification required
 No modification; address < 1000₈ → EXIT
 Address relative to 1000₈ (i.e. 1000₈ ≤ address ≤ 7777₈)
 10--- CW ; relative constant
 No modification; 11000₈ ≤ address ≤ 17777₈ → EXIT
 20--- CW; fixed constant
 No modification; 21000₈ ≤ address ≤ 21777₈ → EXIT
 22---, 23---, 24---, 25---, 26---, or 27--- CW
 No modification; 30000₈ ≤ address ≤ 37777₈ → EXIT
 4---, or 5--- CW.
 60--- CW; fixed temporary
 61--- CW
 62--- CW
 63--- CW
 64---, 65---, or 66--- CW
 67--- CW
 70--- CW
 71--- CW
 No modification; 72000₈ ≤ address ≤ 74777₈ → EXIT
 75--- CW
 76--- CW

		IA	MT		
⑥0	0	SS	MD	0	Address relative $1000_8 - 1000_8 =$ rel. loc. in rtn. \rightarrow Au
	1	SA	MI1	0	
	2	MJ	0	MS	
⑥1	3	SS	MD1	0	10--- CW-10000 ₈ = rel. loc. in rel. const. reg. \rightarrow Au
	4	SA	MI2	0	Rel. loc. + base running add. rel. const. reg. = abs. add. \rightarrow Au
	5	MJ	0	MS	
⑥2	6	SS	MD3	0	20--- CW-20000 ₈ = rel. loc. in fixed const. reg. \rightarrow Au
	7	SA	MI3	0	Rel. loc. + base running add. fixed const. reg. = abs. add. \rightarrow Au
	10	MJ	0	MS	
⑥3	11	SS	MD10	0	60--- CW-60000 ₈ = rel. loc. in fixed temp. reg. \rightarrow Au
	12	SA	MI4	0	Rel. loc. + base running add. fixed temp. reg. = abs. add. \rightarrow Au
	13	MJ	0	MS	
⑥4	14	TP	MC1	Q	Mask (2 digits of "u") \rightarrow Q
	15	QT	A	A	Rel. loc. in pseudo Op. input reg. \rightarrow Au
	16	SA	MC26	0	Rel. loc. + base running add. pseudo Op. input reg. = abs. add. \rightarrow Au
	17	MJ	0	MS	
⑥5	20	TP	MC1	Q	Mask (2 digits of "u") \rightarrow Q
	21	QT	A	A	Rel. loc. in function input region \rightarrow Au
	22	SA	MC25	0	Rel. loc. + base running add. func- tion input reg. = abs. add. \rightarrow Au
	23	MJ	0	MS	
⑥6	24	TP	MC2	Q	Mask (3 digits of "u") \rightarrow Q
	25	QT	A	A	Rel. loc. in non-subs. var. region \rightarrow Au
	26	SA	7	0	Rel. loc. + base running add. non- subs. var. region = abs. add. \rightarrow Au
	27	MJ	0	MS	
⑥7	30	ST	MD15	Q	67--- CW-67000 ₈ = rel. loc. in constant pool \rightarrow Qu
	31	SP	10	17	Base running add. constant pool \rightarrow Au
	32	SA	Q	0	Rel. loc. + base running add. constant pool = abs. add. \rightarrow Au
	33	MJ	0	MS	

⑥8

34	SS	MD16	0
35	SA	MI5	0
36	MJ	0	MS
	CA	MT37	

70---- CW - 70000g = rel. loc. in
working temporary region → Au
Rel. loc. + base running add.
working temp. reg. = abs. add.
→ Au

69

69A

	IA	MU		
0	SS	MD16	0	71--- CW - 70000g = abs. add. → Au
1	MJ	0	MS	
2	RP	[30000]	BR13	No ⇒ alarm 11
3	EJ	FA	MU4	Callword in Op. File III
4	SN	Q	17	- jn + r → Au
5	SA	MU2	0	+ r → Au
6	SA	MU3	0	Address of word following callword in Op. File III → Au
7	TU	A	MU10	
10	TP	[30000]	A	Word following callword in Op. File III → A
11	TJ	MC23	MU14	(A) = IP (14) command (i.e. flagged cross reference)?
12	TP	A	MI	IP (14) instruction to reference other segment → output
13	MJ	0	MR	Exit from add. modification routine
14	SP	A	17	H.S.S. running add. for referenced routine → Au
15	MJ	0	MS	Exit
	CA	MU16		

Modification Constants

	IA	MC	
0	0	77777	0
1	0	77	0
2	0	777	0
3	17	0	0
4	22	0	0
5	45	0	0
6	56	0	0
7	61	0	0
10	63	0	0
11	75	0	0
12	76	0	0
13	77	0	0
14	31	0	0
15	32	0	0
16	33	0	0
17	34	0	0
20	54	0	0
21	55	0	0
22	05	0	0
23	14	0	0
24	57	0	0
25	0	FI	0
26	0	PI	0
	CA	MC27	

Base add. function input region
 ≡ init. add. term. buffer
 Base add. pseudo Op. input region
 ≡ 13th add. term. buffer

	IA	MD	
0	0	1000	0
1	0	10000	0
2	0	11000	0
3	0	20000	0
4	0	21000	0
5	0	22000	0
6	0	30000	0
7	0	40000	0
10	0	60000	0
11	0	61000	0
12	0	62000	0
13	0	63000	0
14	0	64000	0
15	0	67000	0
16	0	70000	0
17	0	71000	0
20	0	72000	0
21	0	75000	0
22	0	76000	0
23	0	77000	0
	CA	MD24	

Explanation of Modification Routine Inputs (MI)

MIO	[0	0	0]	Output = modified word
1	0	[30000]	0	H.S.S. running address for routine
2	0	[30000]	0	Initial running address relative constant region
3	0	[30000]	0	Initial running address fixed con- stant region
4	0	[30000]	0	Initial running address fixed temp- orary region
5	0	[30000]	0	Initial running address working temporary region
6	0	0	0	Temp.

Explanation of Temporary Storage Region (TS)

TS0	0	0	30000	# fixed constants with rtn. in "v"; Codeword to read Termination then
1	0	0	30000	# blks. Term. in codeword position # lines subj. add. modification in "v"; # lines in segment in "v"
2	0[x	xxx] 00	0	# blks. prelude and rtn. - 1 in codeword position; # full blks. seg. + pref. in position
3	0	30000	30000	Routine callword in "u" W/zero fill; H.S.S. running add. rtn. in "v" W/zero fill
4	0	0	30000	# lines to be trans. to drum image in "v"
5	0	0	30000	# fixed const. in image to be trans. to drum in "v"; Add. following run- ning Preface in "v"
6	0	0	30000	# lines prelude and routine in "v"; initial add. running segment in "v"
7	0	30000	0	# lines in Preface in "u"
10	0	30000	0	Segment # in "u"
11	0	0	30000	Address for "IP" jump to next seg- ment in "v"
12	0	0	30000	# lines statements and routines (running program) + 2 in "v"
13	0[x	xxx] 00	0	Count of blks. binary prog. tape in codeword position
14	0	0	[30000]	Index for count of # library rou- tine names in library list
15	0	0	[30000]	Index for # library routines pro- cessed in segment
16	50	000 04	TI	Codeword to read [n] blks. current tape
17	50	001 04	TI	Codeword to read 1 blk. current tape
20	50	004 04	TI	Codeword to read full image load current tape
21	30	000 04	0	Codeword to move forward [n] blks. current tape
22	0	0	0	# library rtns. for problem-1 in "v"
23	0	0	0	Count of blks. advanced on library tape
24	0	0	0	Working temp.
25	0	0	0	Add. running segment in "v"
26	0	0	0	Routine callword (temp 4)

VII. PROGRAM LISTING PHASE

VII. PROGRAM LISTING PHASE

The function of this phase is to provide a record of the Object Program (absolute computer instructions), produced in response to the sentences of the Source Program (pseudo code sentences). The listing gives the absolute instructions which make up each segment of the Object Program, together with the sentence number or library routine name associated with each group of instructions. The instructions are listed four to a line and read from left to right, and down, in order of increasing High Speed Storage address. The first instruction of a routine, i.e., the group of instructions representing one sentence or one library routine, is positioned in the first line, such that, each instruction whose octal address ends in zero, will appear in the leftmost column of instructions in the listing. Each address ending in zero is listed to the left of the associated instruction. The first address of a routine is also listed in this column of addresses on the line with the first instruction. It is enclosed in parenthesis if it does not end in zero.

The listing also includes, in the same format as above, the pool of constants for the program, and the preface and termination instructions for each segment. The variables for the problem are listed in a different format. The symbol for each non-subscripted variable is listed together with its assigned High Speed Storage address. Initially, the symbol for each subscripted variable (array) is listed together with the range of drum addresses assigned to the array. In addition, each subscripted variable is listed in each segment in which it is referenced, together with the range of High Speed Storage addresses assigned to the array for the particular segment.

The title of the program, the subscripted variables on the drum, the non-subscripted variables, and the constant pool, are listed first, in that order.

Then in turn, each segment of the Object Program is listed. Each segment consists of the Preface (if any), the sentences and Library routines, the subscripted variables in High Speed Storage (if any), and the Termination (if any).

The listing is produced on magnetic tape edited for listing on the High Speed Printer. It is produced on Uniservo 7 if 7 Uniservos are being used and on Uniservo 5 if 5 Uniservos are being used. If the listing exceeds an arbitrary 1200 blocks, the current listing tape is terminated at the end of a page, with the statements, CURRENT LISTING TAPE FULL. PUT NEW 1500 FOOT TAPE ON SERVO ---. START TO CONTINUE LISTING., typed on the on-line Flexowriter. This allows the computer operator to change tapes and restart to continue the listing on a new tape. In addition, the statements, MOUNT NEXT LISTING TAPE ON PRINTER. DO NOT CHANGE POSITION OF PAPER., is included on the tape being terminated, together with a Printer Stop symbol. This informs the High Speed Printer operator that the listing is continued on another tape and allows him to mount the tape and continue. The statement, END OF LISTING., and a Printer Stop is included on the final tape of the listing to inform the printer operator of the end of the listing. The order in which the tapes are to be listed, in order to get a continuous listing, is the responsibility of the computer operator.

When the listing is completed the statements, PROGRAM LISTING ON TAPE ---. and END OF COMPILATION., are typed out. The computer then comes to a "56" stop.

The pages of the listing are numbered thru 999, after which the word CONTINUED is used in lieu of a page number.

The instructions of the Program Listing Phase are divided into four

groups. All four groups are read from the UNICODE System Tape into High Speed Storage; Groups II and III are then transferred to the drum. The instructions in Group I remain in High Speed Storage throughout the execution of this phase and consist of constants, temporaries and certain subroutines referenced by the instructions in the other groups.

The Group IV instructions produce the initial part of the listing, consisting of the program title, the subscripted variables on the drum, the non-subscripted variables, and the constant pool. When this part of the listing has been completed, these instructions are overlayed. In listing the subscripted variables on the drum, the information is obtained from the modified Dimension List, which contains the initial drum address and XS3 symbol for each subscripted variable, in order of increasing drum address. The modified Dimension List is assumed to be on the drum when the phase is referenced. In listing the non-subscripted variables, the XS3 symbols for the variables are obtained from the Symbol List, which contains these symbols in order of the increasing High Speed Storage addresses assigned to the variables. The High Speed Storage address associated with the first symbol in the list is obtained from fixed location 00007; the address for each succeeding variable is obtained by adding one to the address of the preceding variable. The Symbol List is read from Uniservo 5 to the List Buffer in this phase. Similarly, the Constant Pool, containing the constants in order of their increasing High Speed Storage address, is read from Uniservo 5 to the Dimension List region in the core. The High Speed Storage address of the first constant is obtained from fixed location 00010 and that of each succeeding constant is obtained by adding one to the address of the preceding constant. The program title is listed just as it appears on the UNICODE (source) Program Tape; hence, only

printable High Speed Printer characters should be used in the title.

The Group II instructions, lists, etc., initially overlay the Group IV instructions and, thereafter, overlay the Group III instructions and lists. The Group II instructions are read from the drum to core whenever a new segment is to be listed and, finally, when the listing phase is to be terminated. The instructions in this group build Op. File IV for the segment to be listed and store it on drum; then they are overlaid by the Group III instructions, lists, etc. The information to build Op. File IV is obtained from Op. File III for each segment and from the Sentence Number List, which is produced by the Processor Phase and stored on the drum as input to this phase. Op. File III for each segment is read from Uniservo 5 to the File Buffer. The Group II instructions also terminate the final listing tape, rewind all tapes not yet rewound, and produce the Flexowriter typeouts at the completion of the phase.

The Group III instructions produce the listing of the segments. The Preface and Termination instructions for the segment to be listed are obtained, for listing, from Uniservo 5 following the Op. File III for the segment. The Preface is read from Uniservo 5 to the Input Buffer. The initial High Speed Storage address for the Preface is obtained from the seventh word of the Segment Label Block on the Object Program Tape, and the number of lines in the Preface is obtained from the eighth word. With these as inputs, the Preface is edited and written on the listing tape in the prescribed format, by an editing routine which is common for the Preface, Termination, Constant Pool, Sentences, and Library Routines. In listing the sentences and the library routines, the number of routines in the segment being listed is obtained from Temporary (CT5) which is set up by the routine which builds Op. File IV for the segment. The XS3 sentence number or library routine name for

a routine to be listed, the number of lines in the routine, and the initial High Speed Storage address of the routine are obtained from Op. File IV and provided as inputs to the common editing routine which edits and writes each routine on the listing tape. The sentences and library routines appear in Op. File IV in order of increasing High Speed Storage address. The Termination is read from Uniservo 5 to the Input Buffer prior to the listing of the subscripted variables in the core. In listing these variables the High Speed Storage address, in the segment being listed, is obtained in order, from the instructions of the Termination. The modified Dimension List is then searched for the drum address in order to find the XS3 symbol for the variable. The variables are then edited and listed in the prescribed format by an editing routine used in common to list the subscripted variables on the drum and in the core. The Termination is listed by sections, each representing one block of the Termination. The total number of lines in the Termination is obtained from the eighth word of the Segment Label Block. The initial High Speed Storage address of each section is merely the initial address of the Termination Buffer which is a constant. The number of lines in each section is 170 octal except for the last section which is the number of lines in the partial block remaining. Again, the common editing routine is used.

Because of the overlaying involved in the execution of the Program Listing Phase, considerable care should be exercised in making changes in the addresses or lengths of routines, lists, etc.

OP. FILE IV

Op u v

[XS3	Sentence	Number]
00	[Number of lines]	[H.S.S. Address]	
[XS3	Sentence	Number]
00	[Number of lines]	[H.S.S. Address]	
[XS3	Sentence	Number]
00	[Number of Lines]	[H.S.S. Address]	

In order of the increasing magnitude of the H.S.S. Address Entries



Format of entry for statements, equations, and Pseudo Ops.

etc.			
[XS3	Library Routine Name]
[74]	[Number of lines]	[H.S.S. Address]	

Format of entry for Library Routines

Where:

Number of lines = The number of instructions, including constants and temporaries, in the routine in the Object Program associated with the preceding XS3 sentence number or name.

H.S.S. Address = The High Speed Storage running address of the routine in this segment of the Object Program.

The Op. File IV for each segment is built by the Program Listing Phase just prior to the listing of the segment. The information for the list is obtained from Op. File III for the segment and from the Sentence Number List.

Entries are made in Op. File IV for only those routines which are included in the segment to be listed. Sentence numbers, for which a callword appears in Op. File III followed by an "Interpret" instruction, are omitted from Op. File IV. The "Interpret" instruction indicates the routine is in another segment but is referenced from the segment being listed.

The Library Routine entries have a 74₀ in the Op. code of the second word to indicate that they are Library Routine entries. All other entries have a 00.

Modified Dimension List

00		[Drum Address]	0 0 0 0 0
		[XS3 Symbol for array]
00		[Drum Address]	0 0 0 0 0
		[XS3 Symbol for array]
		etc.	
~~~~~			
		[ XS3    Symbol for array	]
0		[ Drum Address]	0 0 0 0 0

In order of the increasing magnitude of the drum address entries



← Last entry in list might =100000₈.

Where:

Drum Address = Initial address of array on drum during running of Object Program.

XS3 symbol = XS3 symbol for subscripted variable (array) to which preceding drum address applies.

The modified Dimension List is built by a routine which operates during the Allocation Setup Phase. Information to build the list is obtained from the original Dimension List in the Combination List, which is still available at this time.

Fixed location 00010 is changed at the time the modified Dimension List is built to describe this new list.

The last entry in the list must always be the address following the last address of the last array on drum. If, therefore, the last address of the last array were  $77777_8$ , the next address would be  $100000_8$ . Although this is not a legitimate address, in this case it would have to be included as the last entry in the list.

### Sentence Number List

(ND = Regional Address of List on Drum)

Op.            u            v

XS3 Sentence Number

XS3 Sentence Number

etc.

ND - Section for sentence numbers associated with 26---, 27---, and 22--- type callwords. (Maximum of  $512_{10}$  such callwords)

XS3 Sentence Number

XS3 Sentence Number

etc.

ND1000₈ - Section for sentence numbers associated with 24--- type callwords. (Maximum of  $512_{10}$  such callwords)

XS3 Sentence Number

XS3 Sentence Number

etc.

ND2000₈ - Section for sentence numbers associated with 25--- type callwords. (Maximum of  $512_{10}$  such callwords)

XS3 Sentence Number

XS3 Sentence Number

etc.

ND3000₈ - Section for sentence numbers associated with 4--- type callwords. (Maximum of  $64_{10}$  such callwords)

XS3 Library Routine Name

XS3 Library Routine Name

etc.

ND3100₈ - Section for names associated with 5--- type callwords. (Maximum of  $512_{10}$  such callwords)

The Sentence Number List is built by a routine which operates during the Processor Phase, where the Prelude of each routine is still available. The callword of each routine, and the associated XS3 sentence number or Library Routine name, are obtained from the Prelude of the routine.

The entries in each section of the list are stored within the section relative to the last three octal digits of the callword, except for 4---- and 5---- type callwords. For the 4xxx - and 5xx-- type callwords, the digits marked "X" are used.

The sections of the list always remain at the same relative distances from the beginning of the list, as shown on the preceding diagram; hence the list is always  $4100_8$  locations long.

SYMBOL LIST FORMAT

Op.	u	v
XS3	SYMBOL	FOR VARIABLE
XS3	SYMBOL	FOR VARIABLE
XS3	SYMBOL	FOR VARIABLE

In order of the Increasing H.S.S. Addresses

Where:

XS3 SYMBOL = XS3 Symbol for each of the non-subscripted variables of the problem

This list is built and written on Uniservo 5 by routines which operate during the "End of Tape" generation phase. The list contains the XS3 symbols for all the functions (66--- callwords), floating point non-subscripted variables (65--- callwords), and fixed point variables (64--- callwords) of the problem. The symbols are in the list in order of the increasing High Speed Storage addresses assigned to the variables.

Program Listing Phase Setup Block  
Regional Assignments:

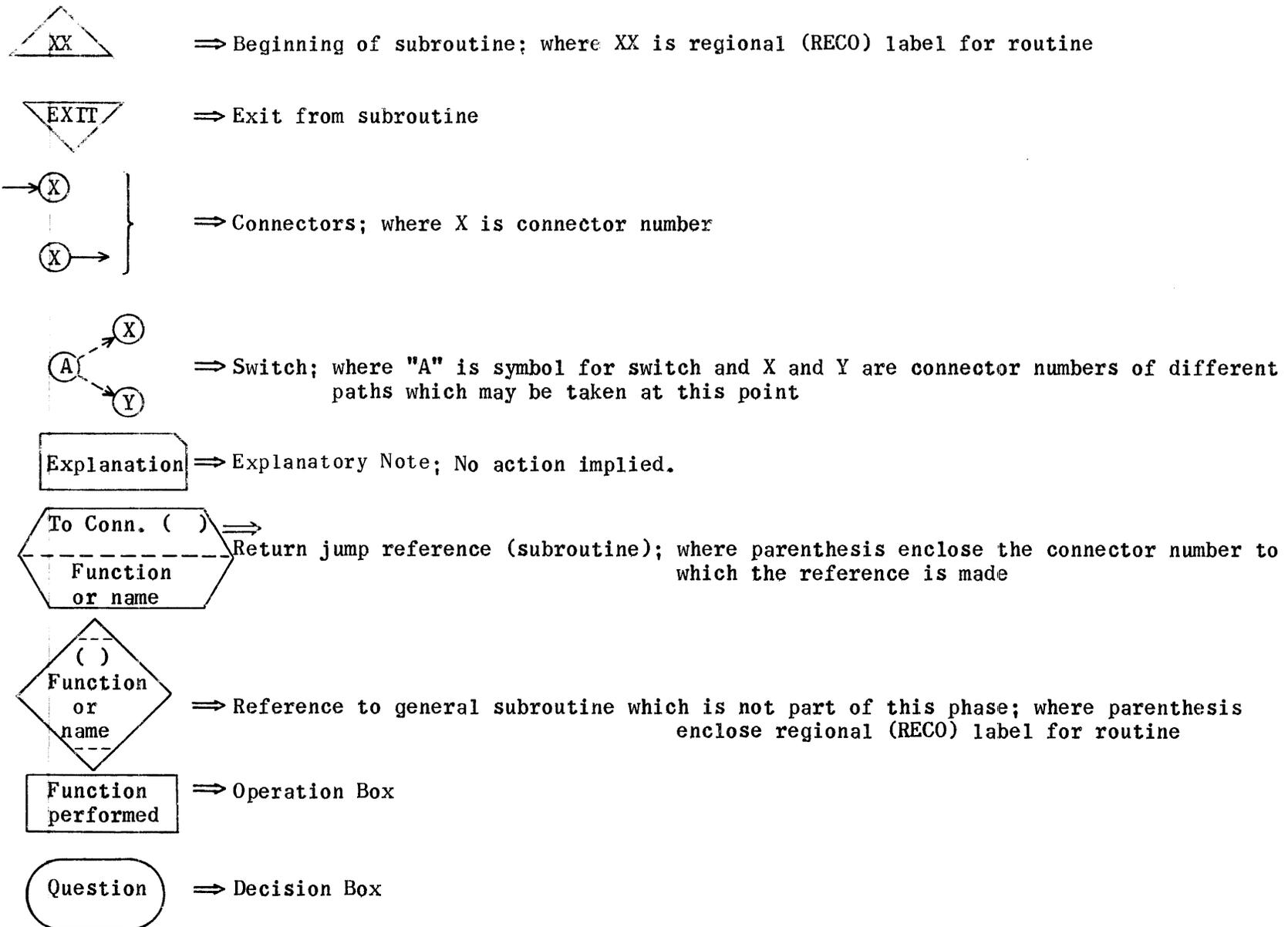
RE TN20  
RE TH21  
RE UP421  
RE FP653  
RE PK2547  
RE LS7230  
RE LT7260

Setup Block

	IA	LS		
0	SP	TN	0	
1	ZJ	LS2	LS3	(A) = zero? $\Rightarrow$ 5 servos; (A) $\neq$ 0 $\Rightarrow$ 7 servos
2	RA	LT11	LS24	Adv. servo # in printout by 3 to set obj. prog. tape # = 6
3	TP	LT	UP3	
4	RJ	UP2	UP	Typeout: COMPUTER CODING PRODUCED ON TAPE 3 or 6
5	TP	LT12	UP3	
6	RJ	UP2	UP	Typeout: IF PROGRAM LISTING IS NOT DESIRED, SET A NOT = 0. START.
7	SP	LS25	0	Set A = 0.
10	MS	0	LS11	
11	ZJ	LS17	LS12	Program Listing desired?
12	TP	LS26	TH3	
13	RJ	TH2	TH	Read program listing phase from servo 1 to core
14	TP	LS27	TH3	
15	RJ	TH2	TH	Rewind servo 1
16	MJ	0	PK	Jump to program listing phase
17	TP	LS27	TH3	
20	RJ	TH2	TH	Rewind servo 1
21	TP	LT25	UP3	
22	RJ	UP2	UP	Typeout: COMPILATION COMPLETED.
23	MS	0	LS23	
24	0	0	300	
25	0	0	0	
26	50	01201	FP	Tape codeword to read listing phase to core
27	10	1	0	Tape codeword to rewind servo 1
	CA	LS30		

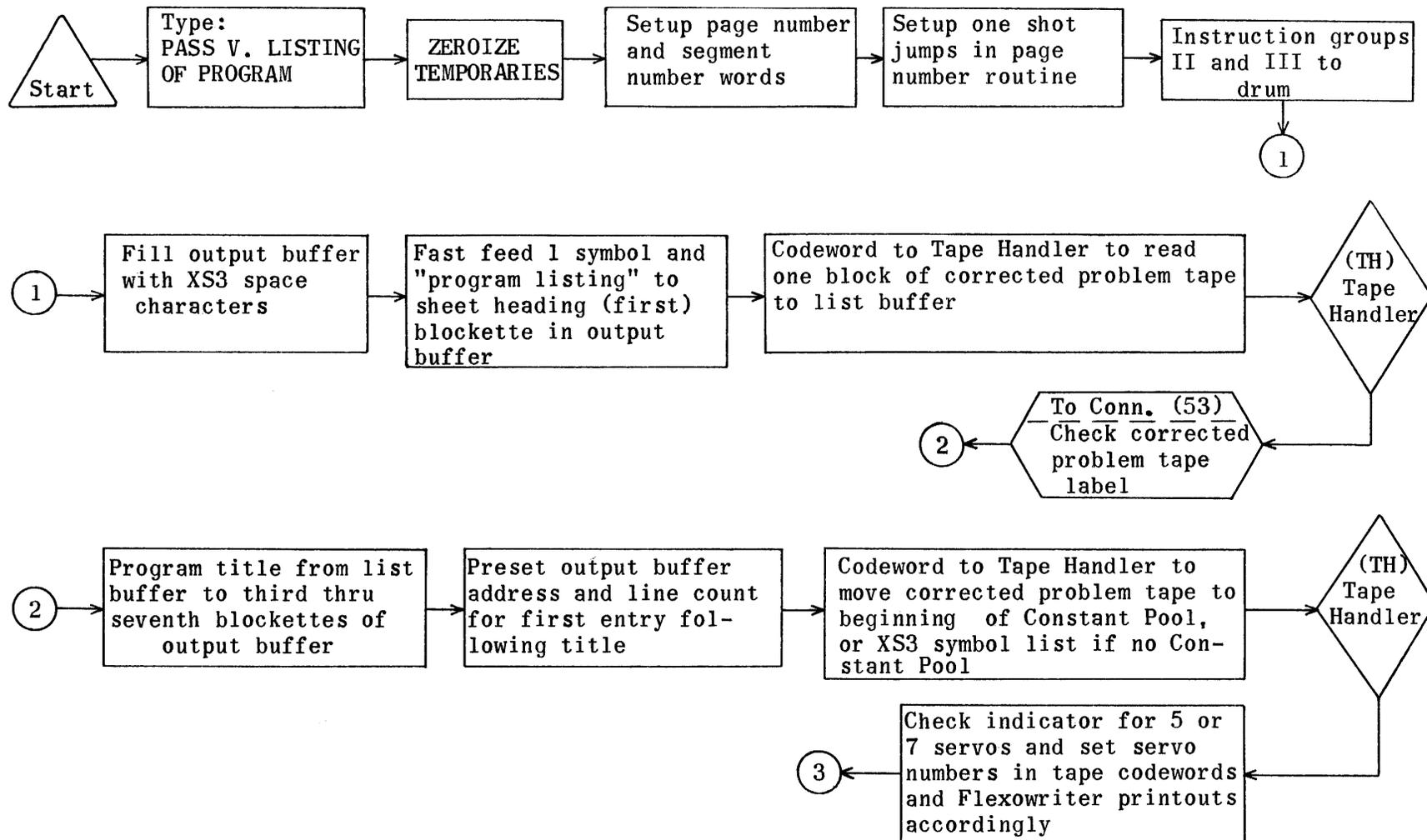
Listing Setup Typeout

0	IA	LT							
0	0	LT1	11						
1	01	01010	10101	△	△	△	△	△	△
2	01	01010	10101	△	△	△	△	△	△
3	01	01010	10101	△	△	△	△	△	△
4	01	26514	75267	△	C	O	M	P	U
5	66	30540	12651	T	E	R	△	C	O
6	27	34503	20152	D	I	N	G	△	P
7	54	51276	72630	R	O	D	U	C	E
10	27	01515	00166	D	△	O	N	△	T
11	24	52300	10622	A	P	E	△	3	.
12	0	LT13	12						
13	34	31015	25451	I	F	△	P	R	O
14	32	54244	70146	G	R	A	M	△	L
15	34	65663	45032	I	S	T	I	N	G
16	01	34650	15051	△	I	S	△	N	O
17	66	01273	06534	T	△	D	E	S	I
20	54	30272	10165	R	E	D	,	△	S
21	30	66012	40150	E	T	△	A	△	N
22	51	66017	60322	O	T	△	=	O	.
23	01	01656	62454	△	△	S	T	A	R
24	66	22777	77777	T	.	77	77	77	77
25	0	LT26	7						
26	01	01010	10101	△	△	△	△	△	△
27	01	01010	10101	△	△	△	△	△	△
30	01	01010	10101	△	△	△	△	△	△
31	01	26514	75234	△	C	O	M	P	I
32	46	24663	45150	L	A	T	I	O	N
33	01	26514	75246	△	C	O	M	P	L
34	30	66302	72277	E	T	E	D	.	77
	CA	LT35							

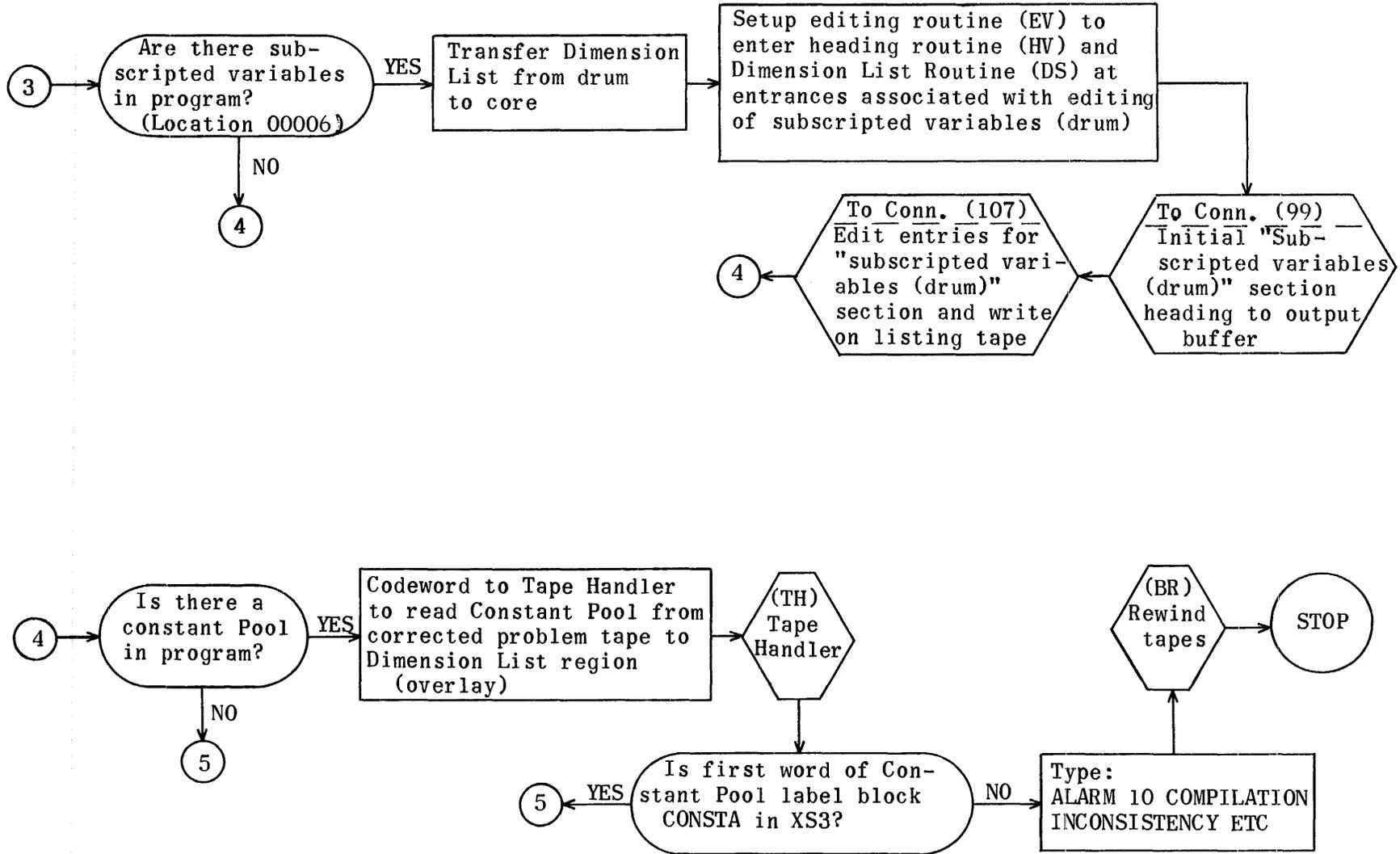


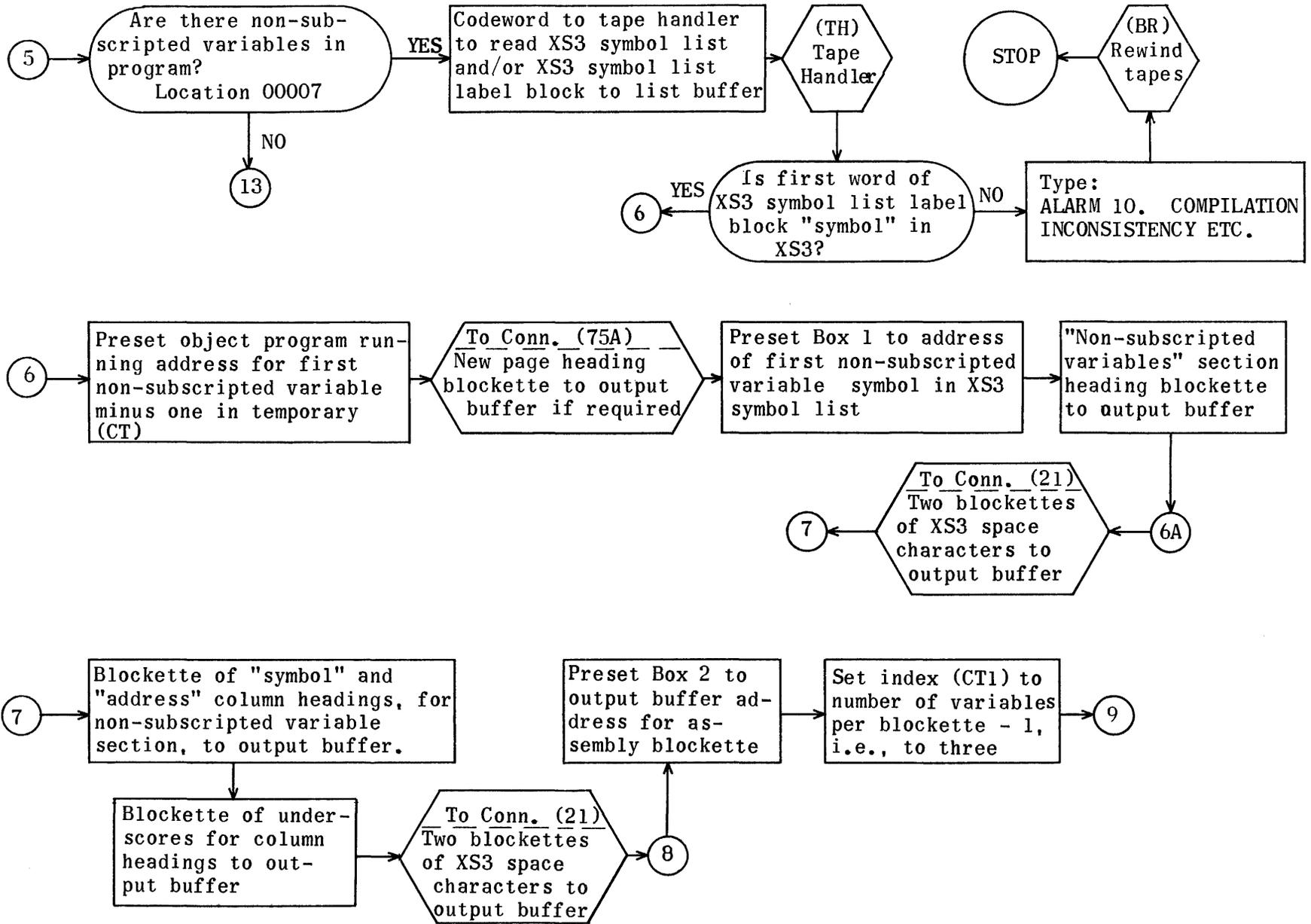
Key to Program Listing Phase Flow Chart

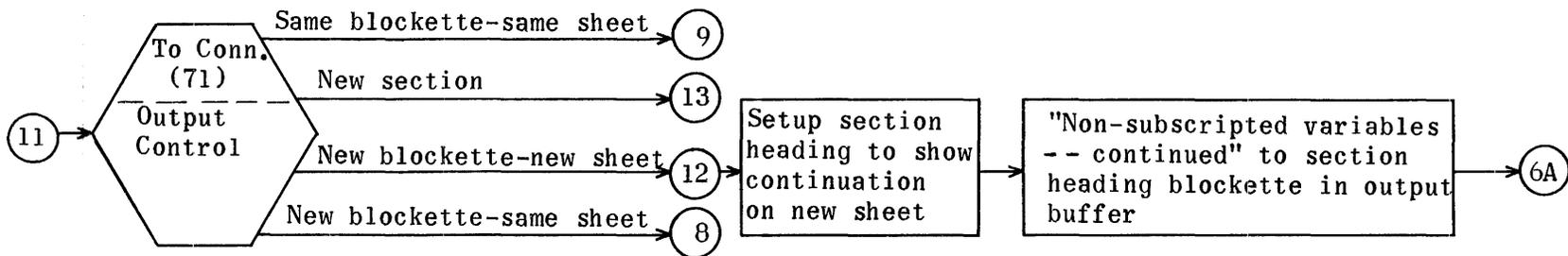
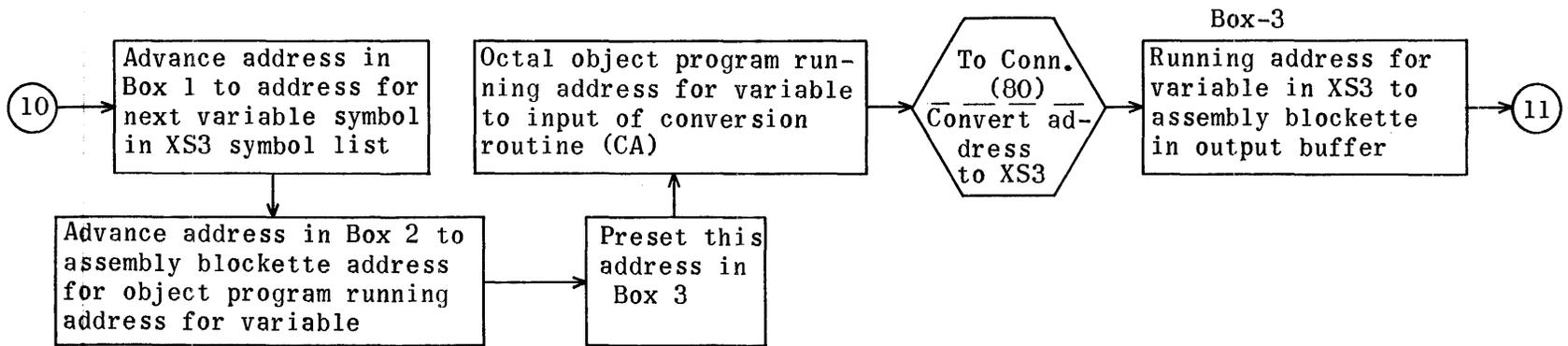
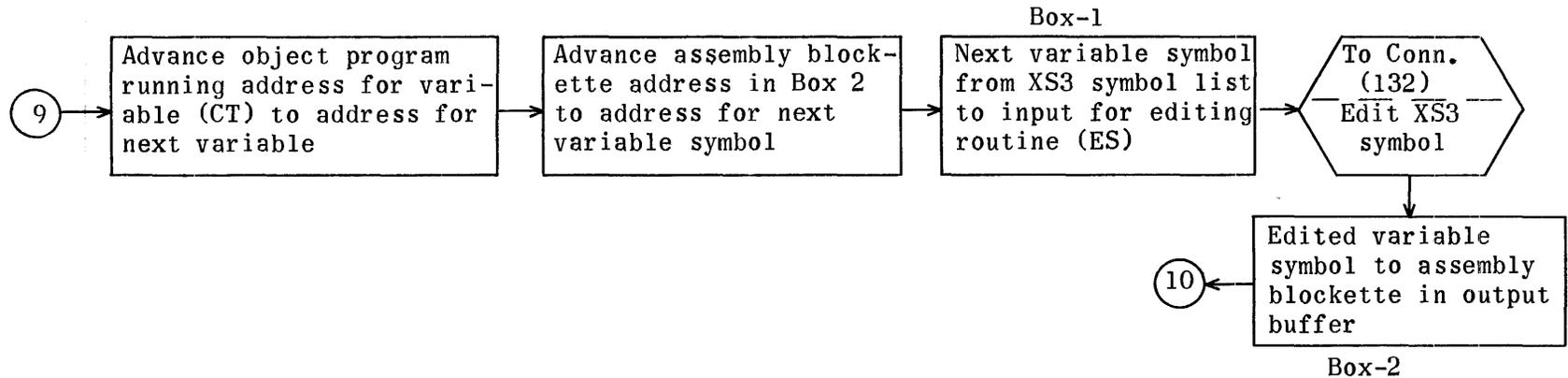
Program Listing Phase

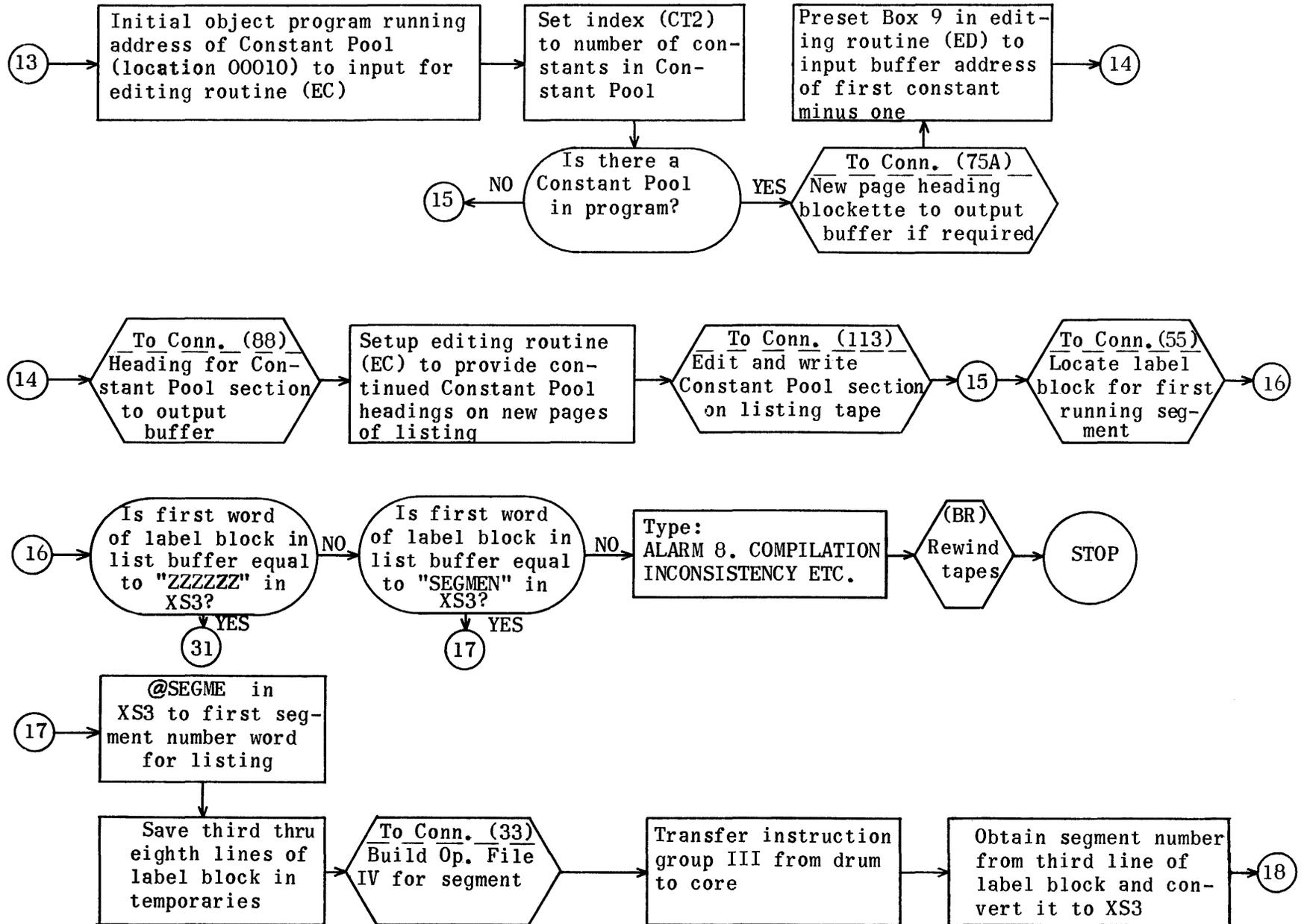


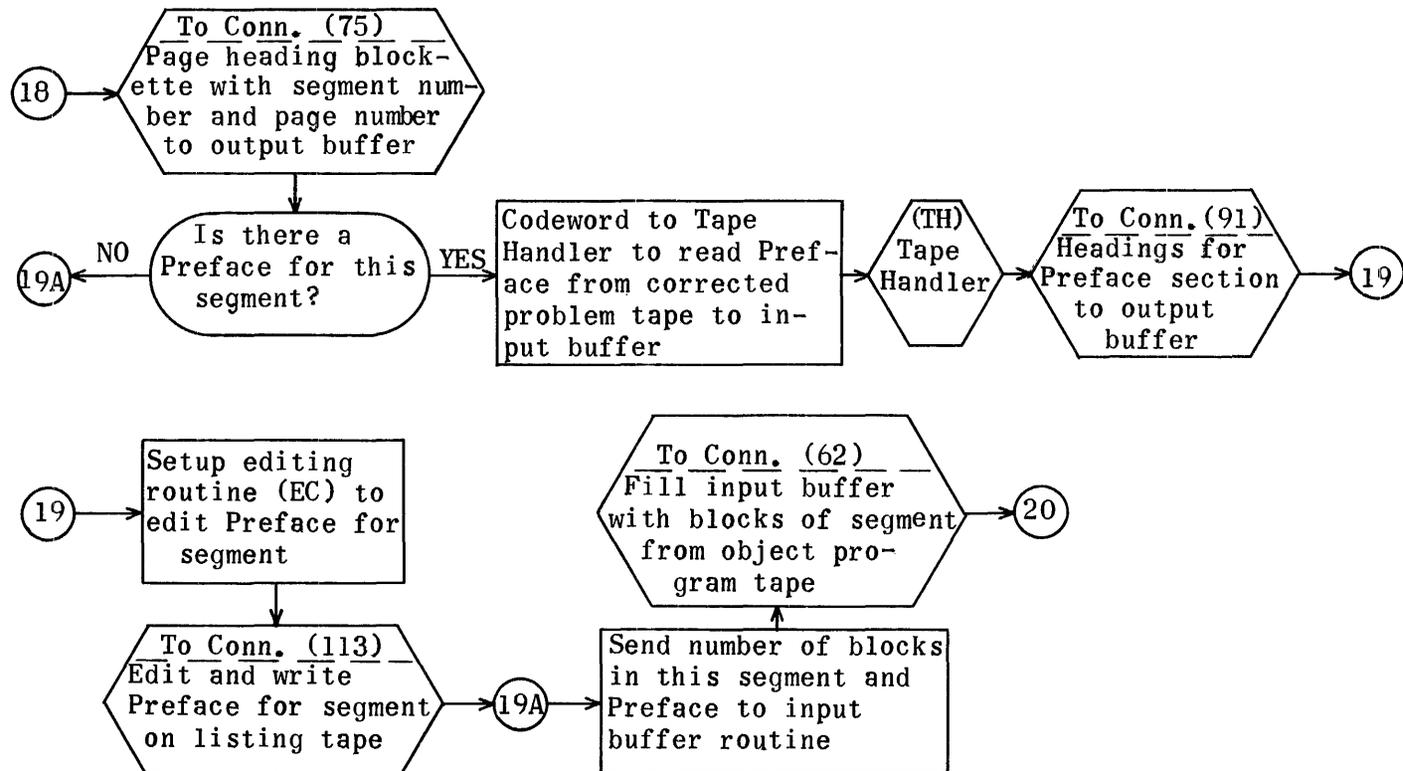
1762

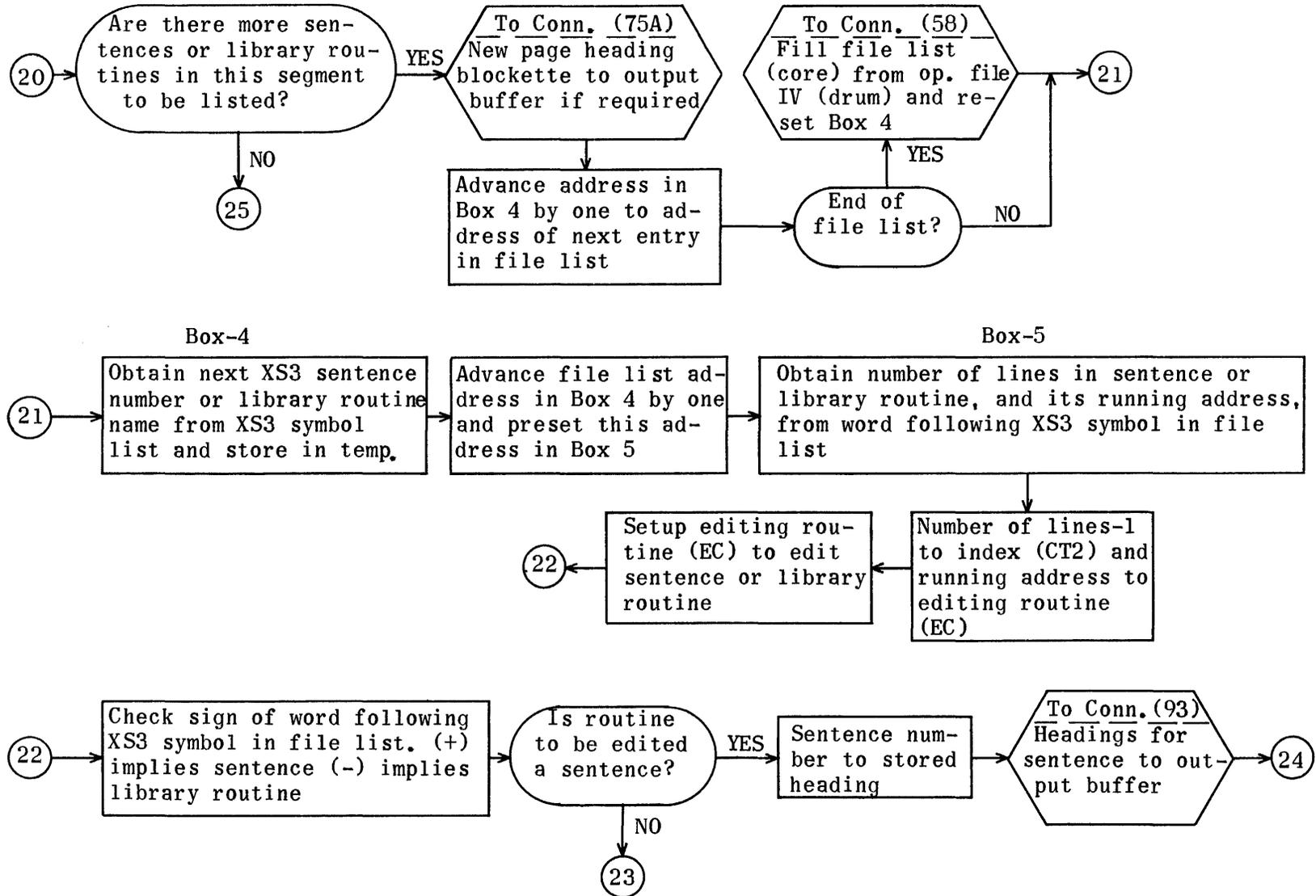


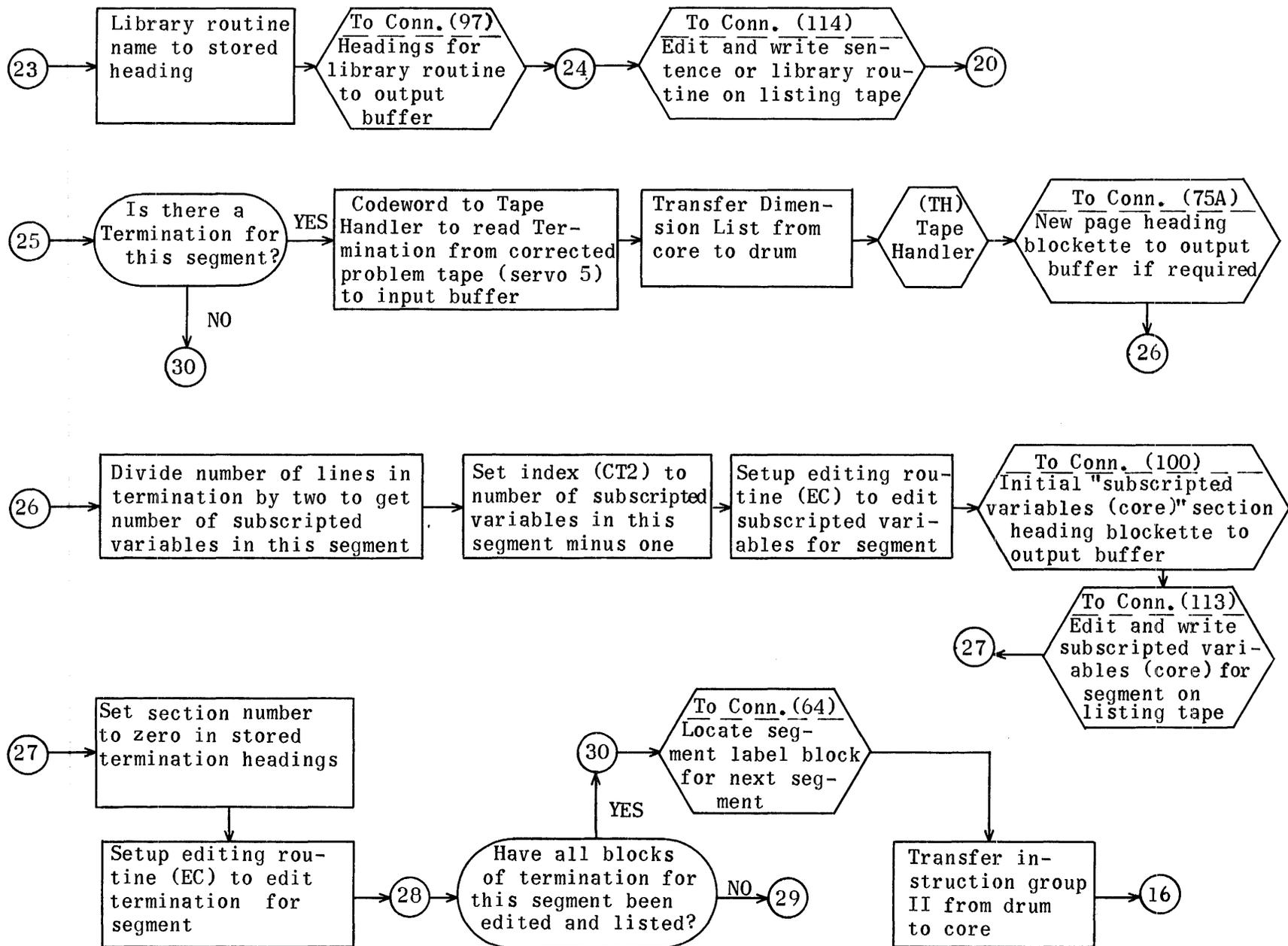


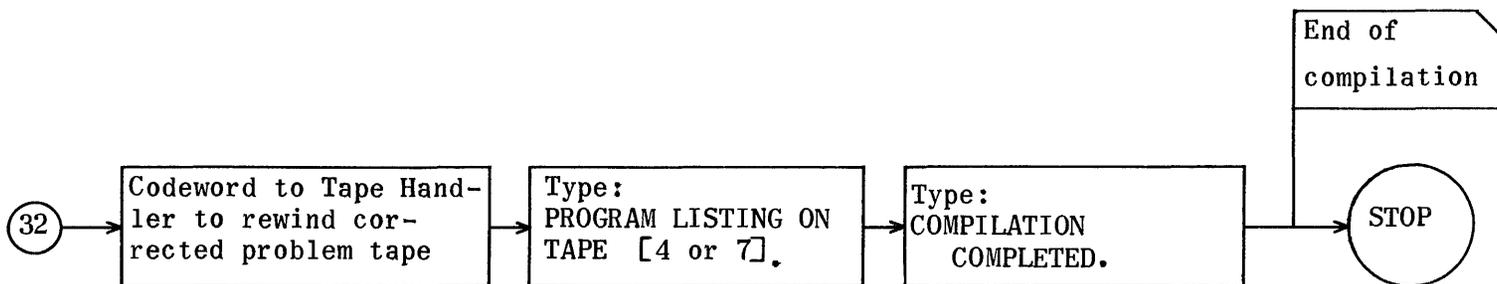
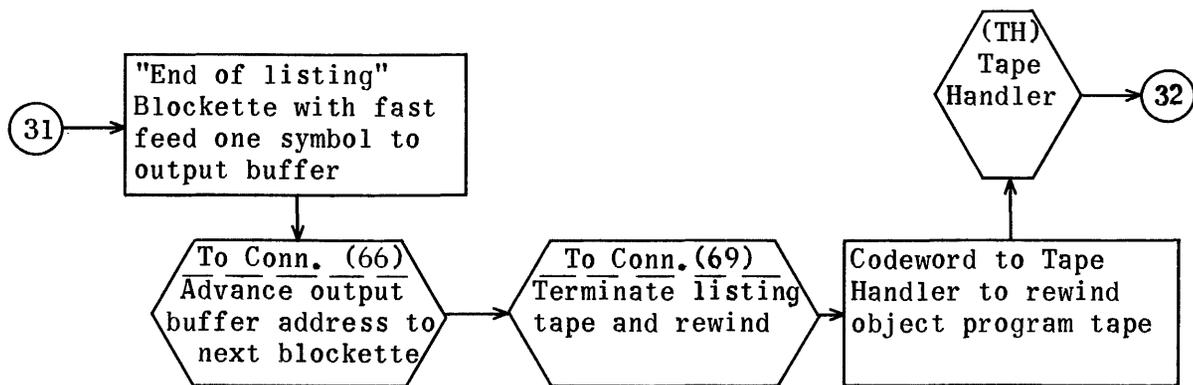
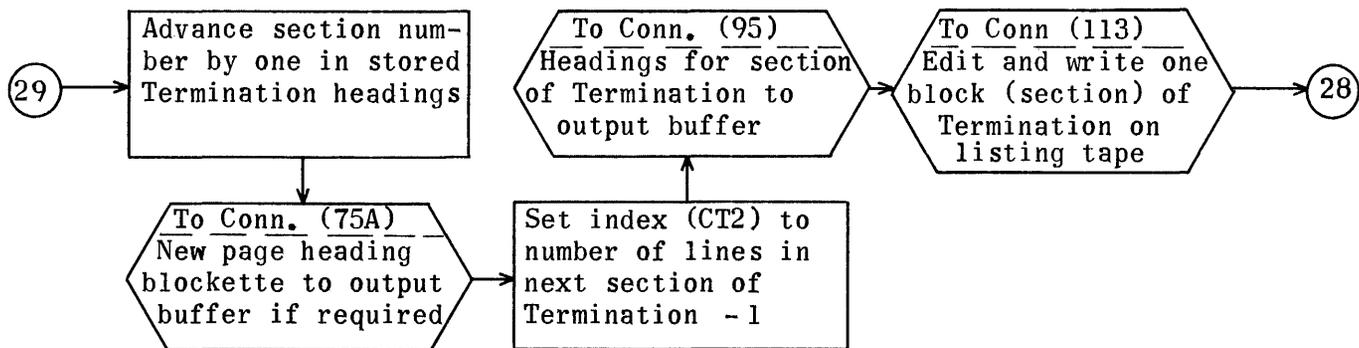




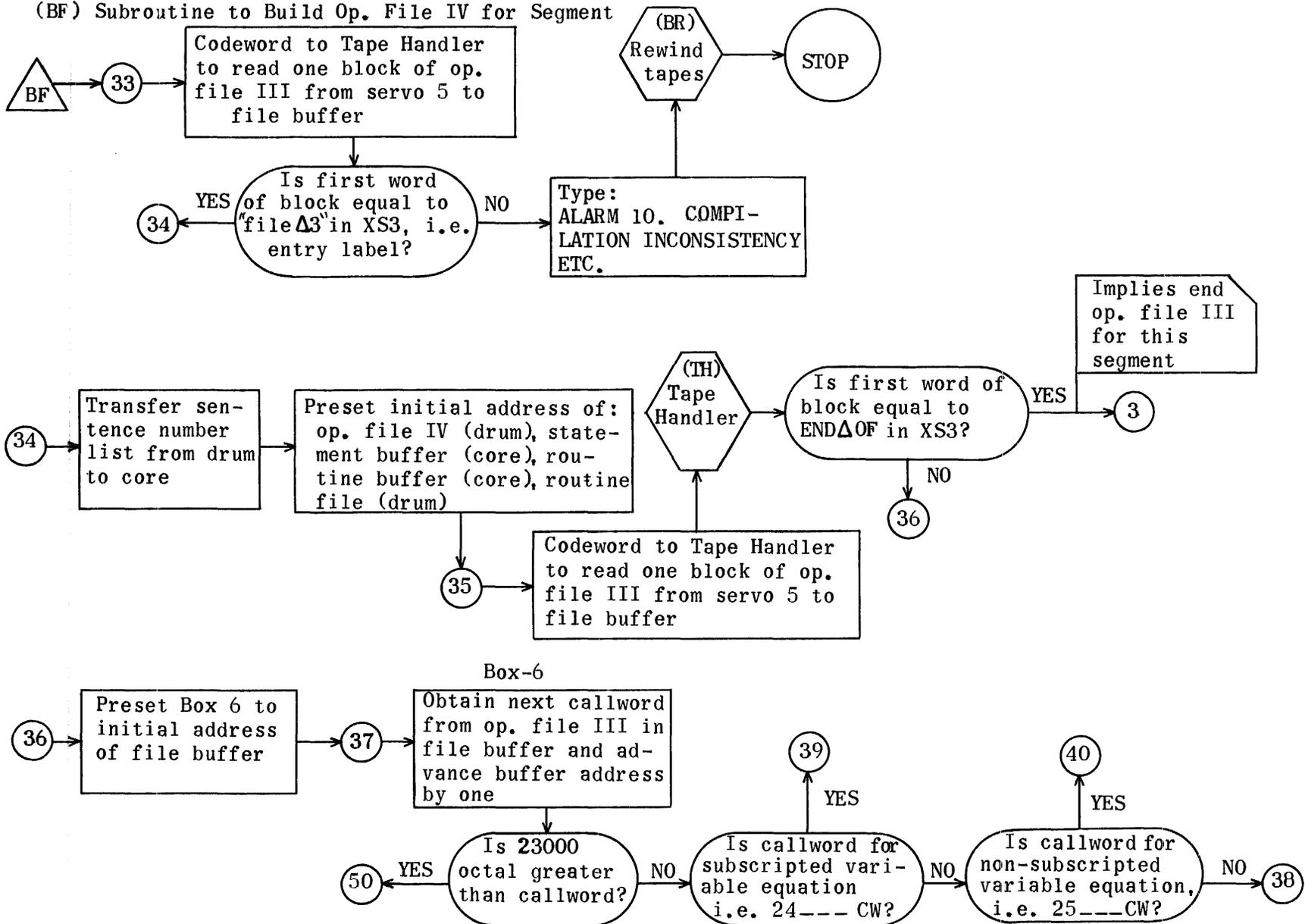


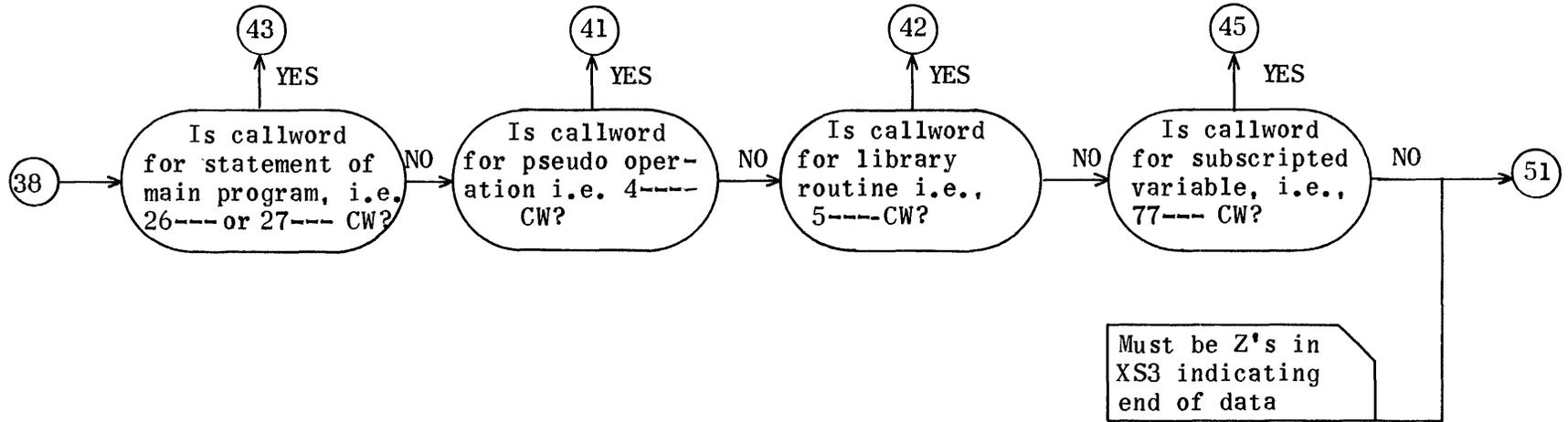




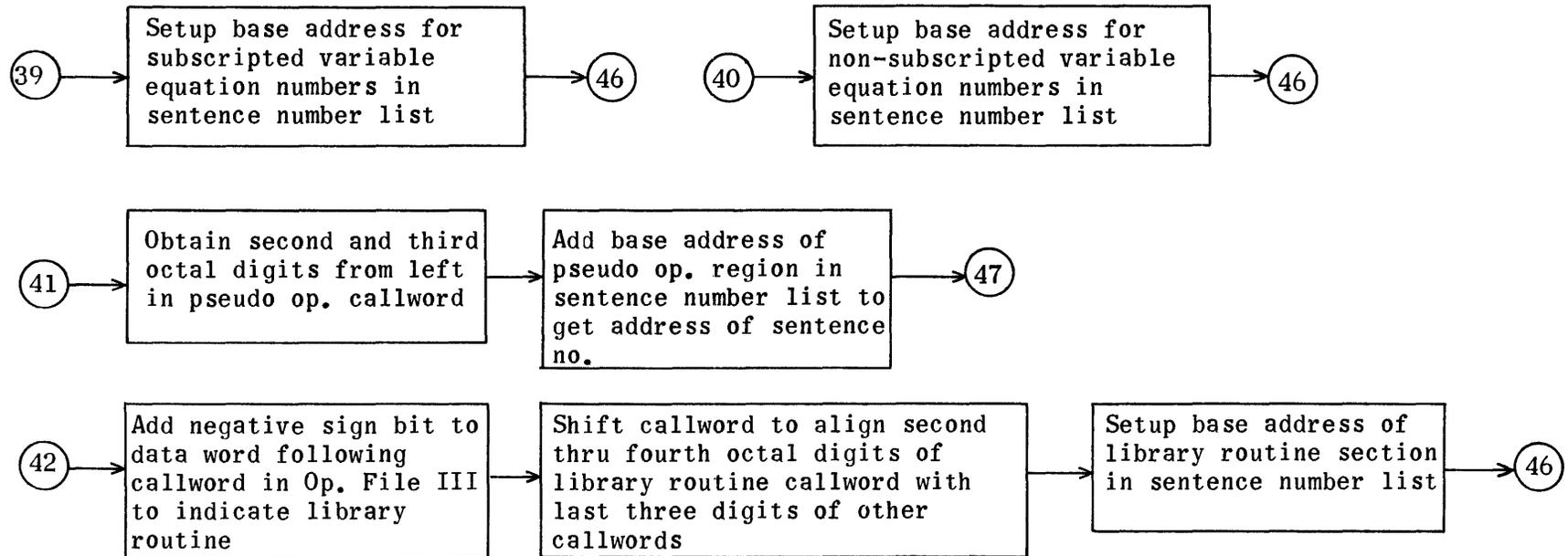


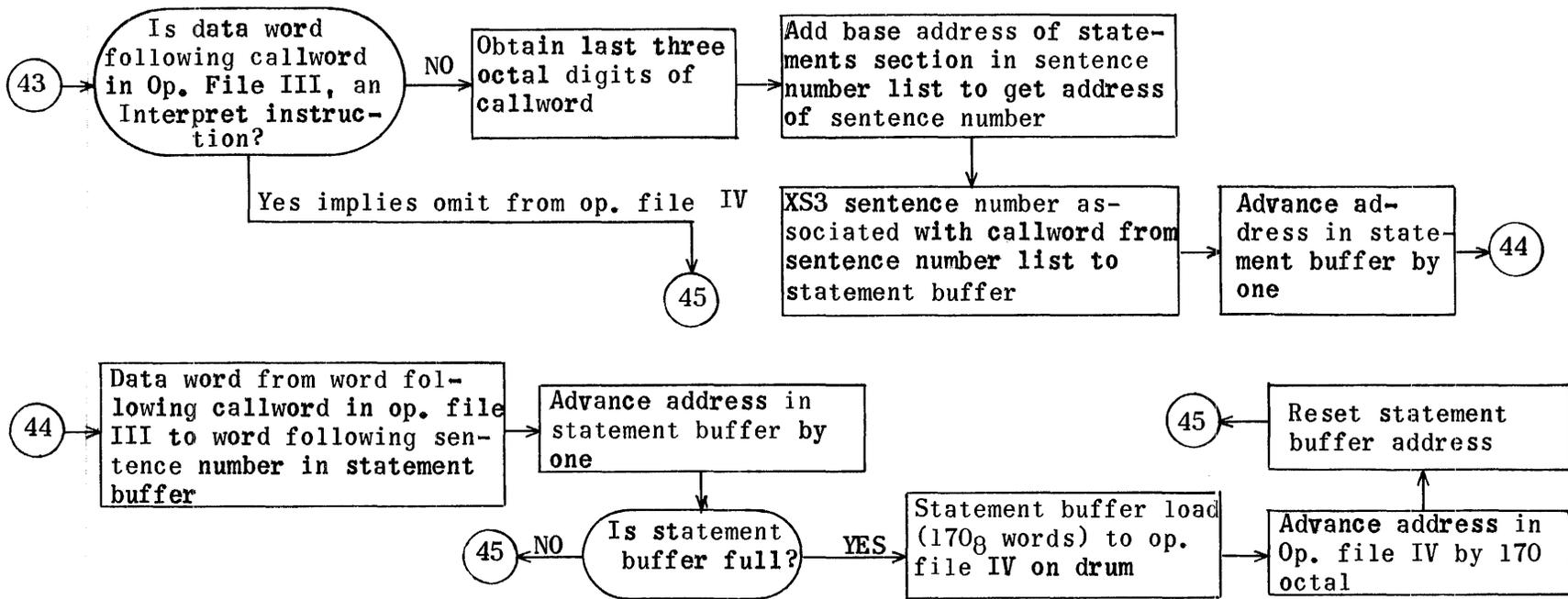
(BF) Subroutine to Build Op. File IV for Segment

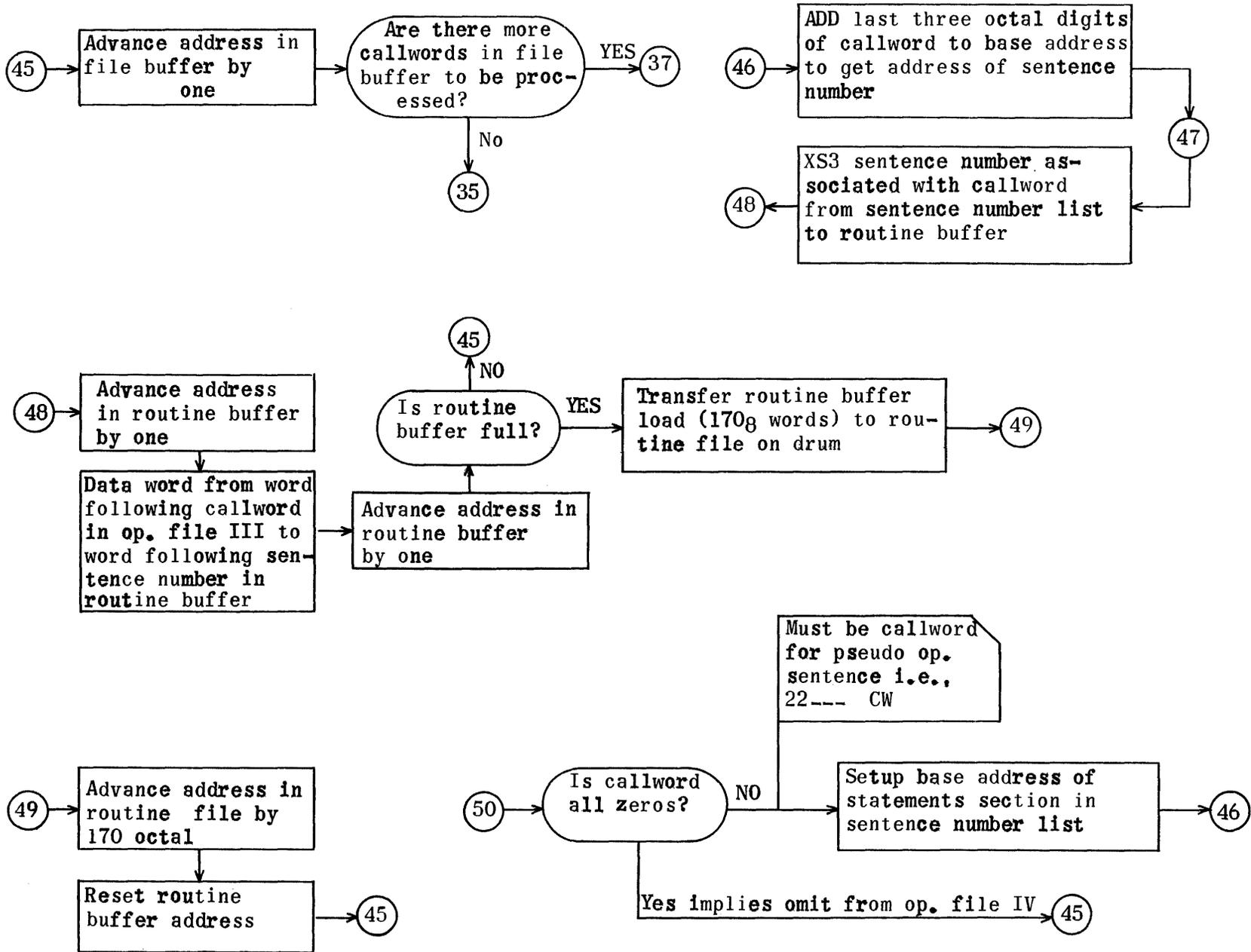


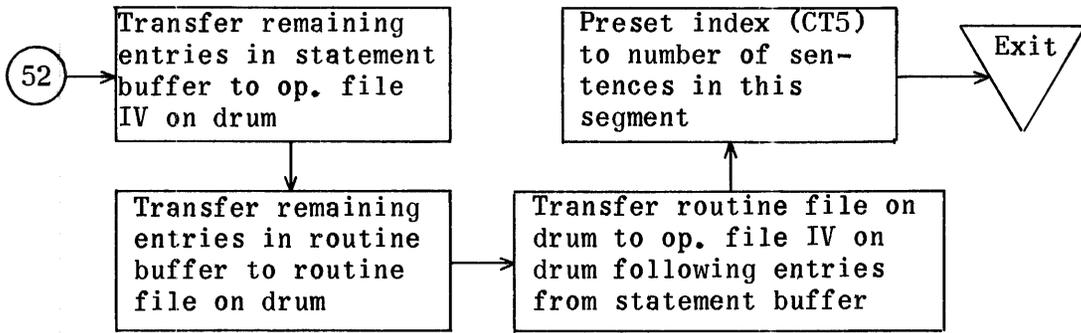
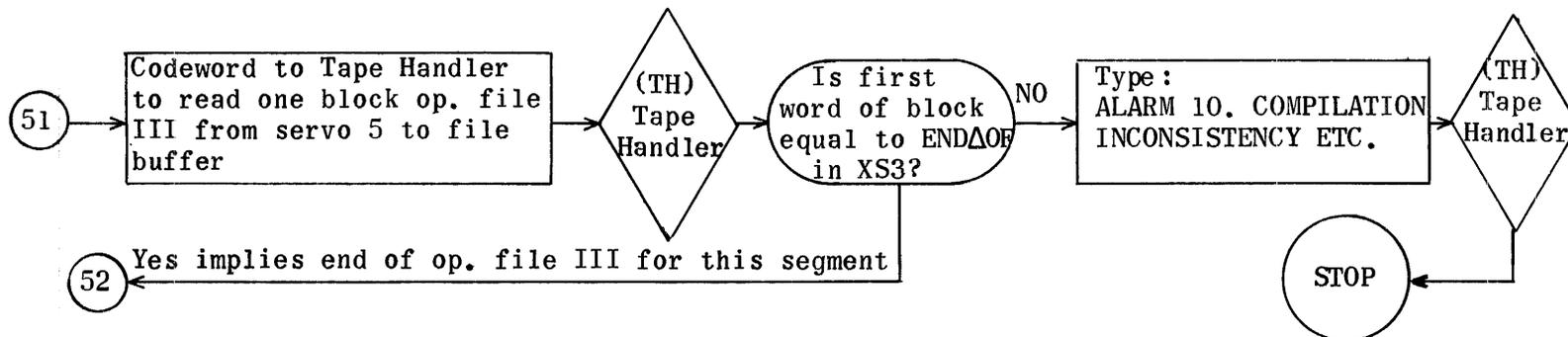


1772



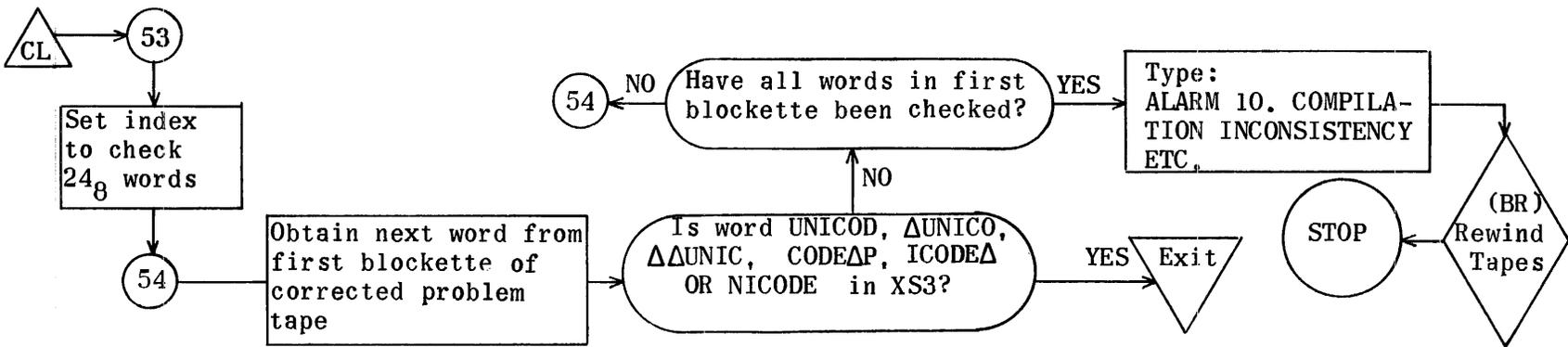




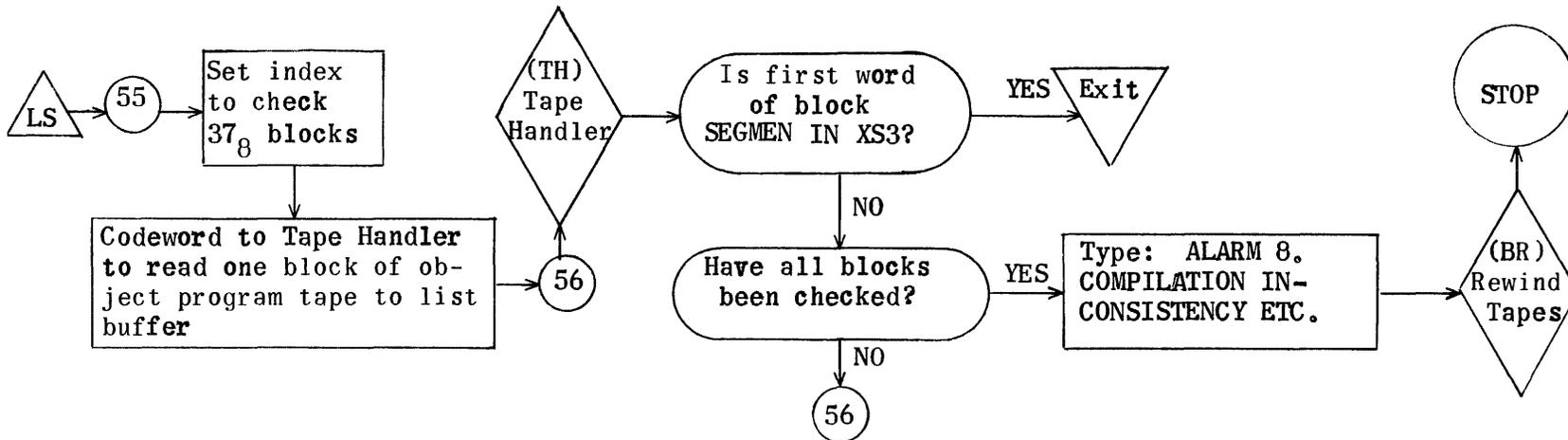


1775

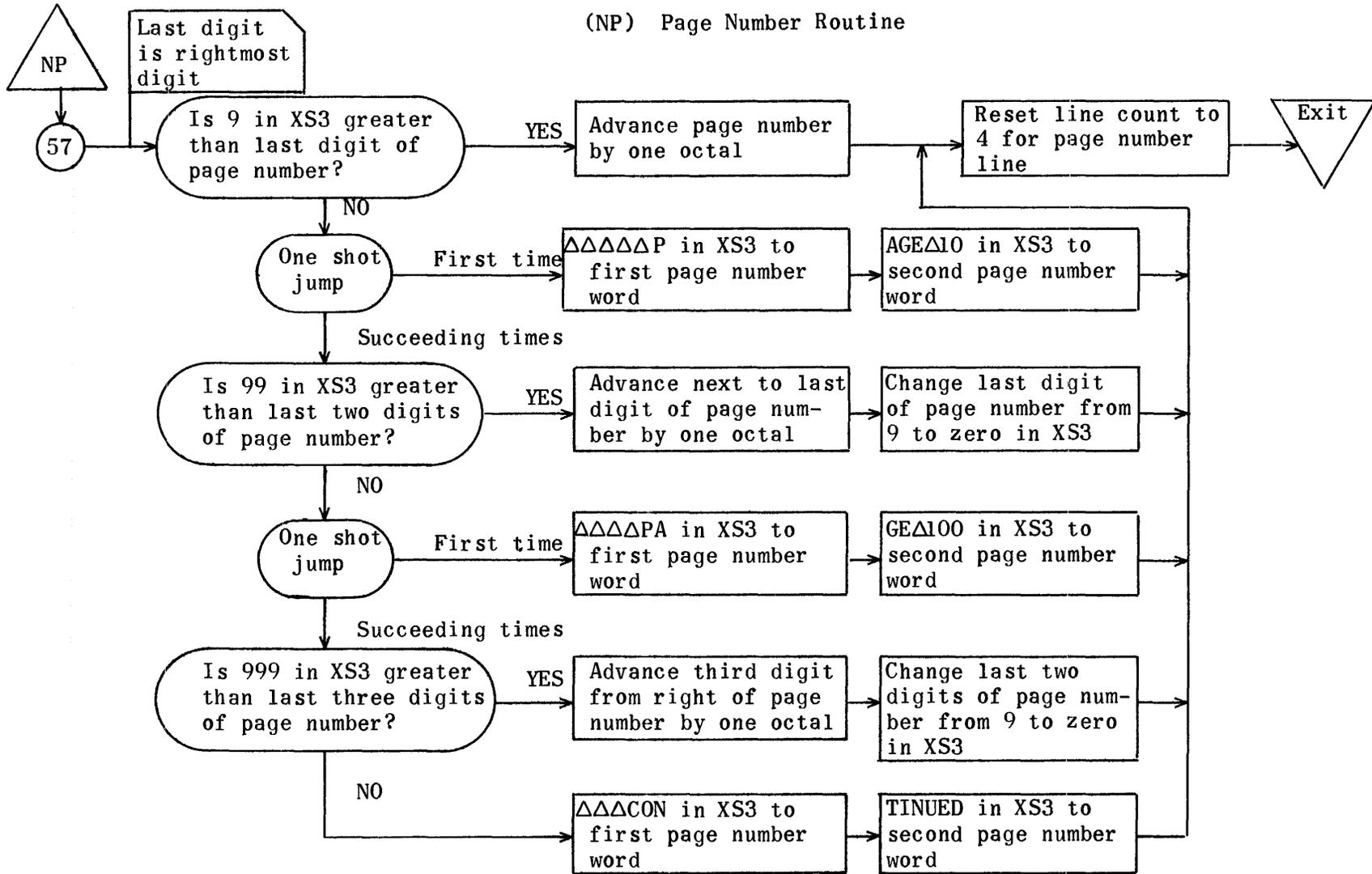
(CL) Check label on corrected problem tape  
(i.e., check for UNICODE PROGRAM anywhere in first blockette)



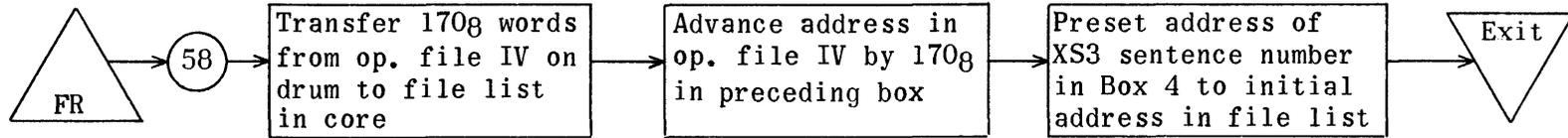
(LS) Locate segment label block for first segment on object program tape



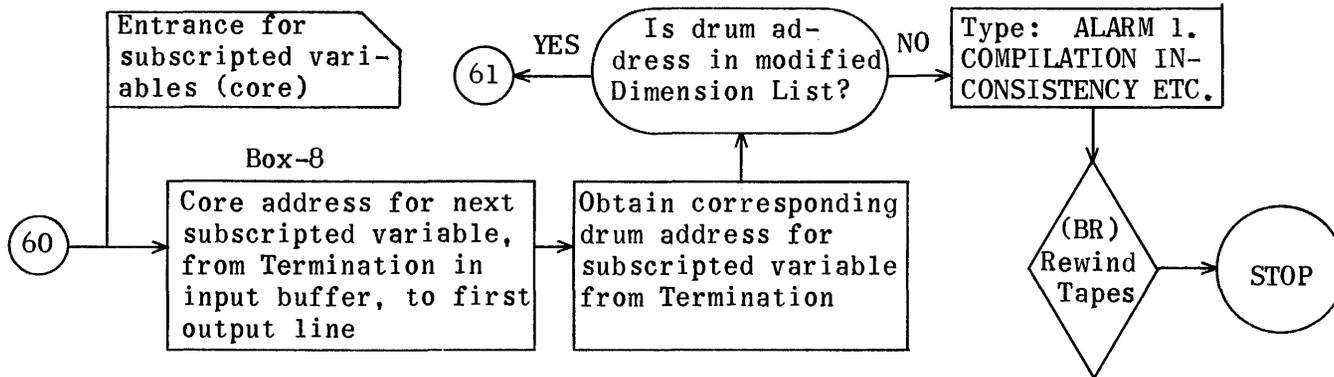
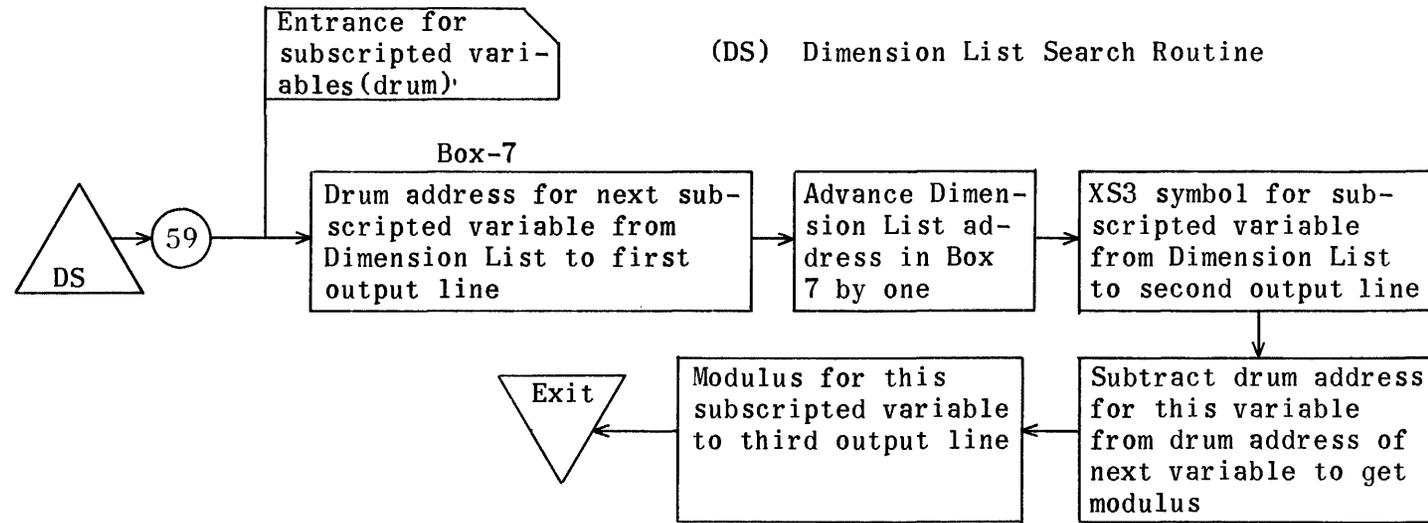
(NP) Page Number Routine

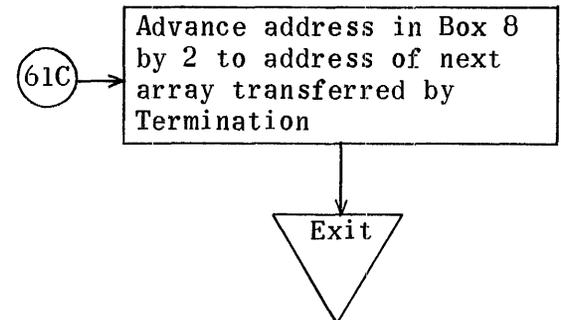
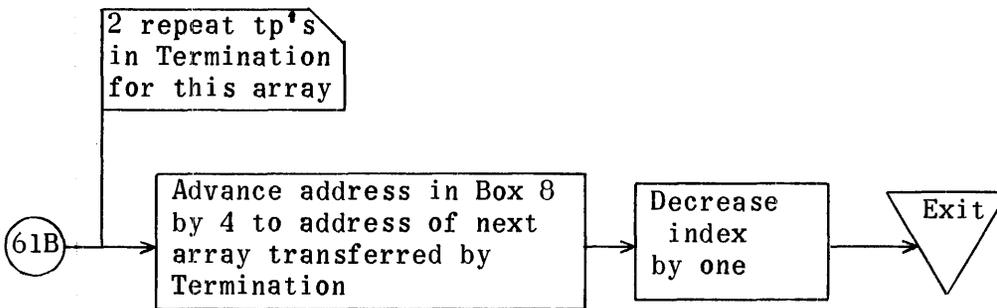
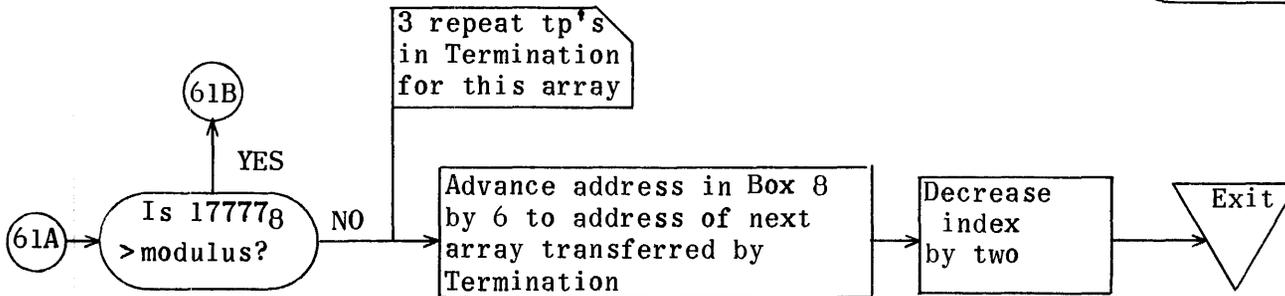
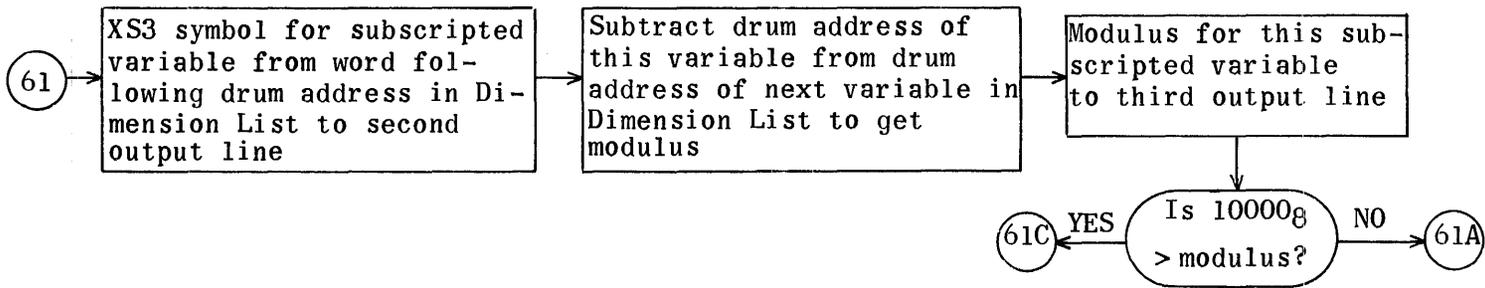


(FR) Op. File III Control Routine



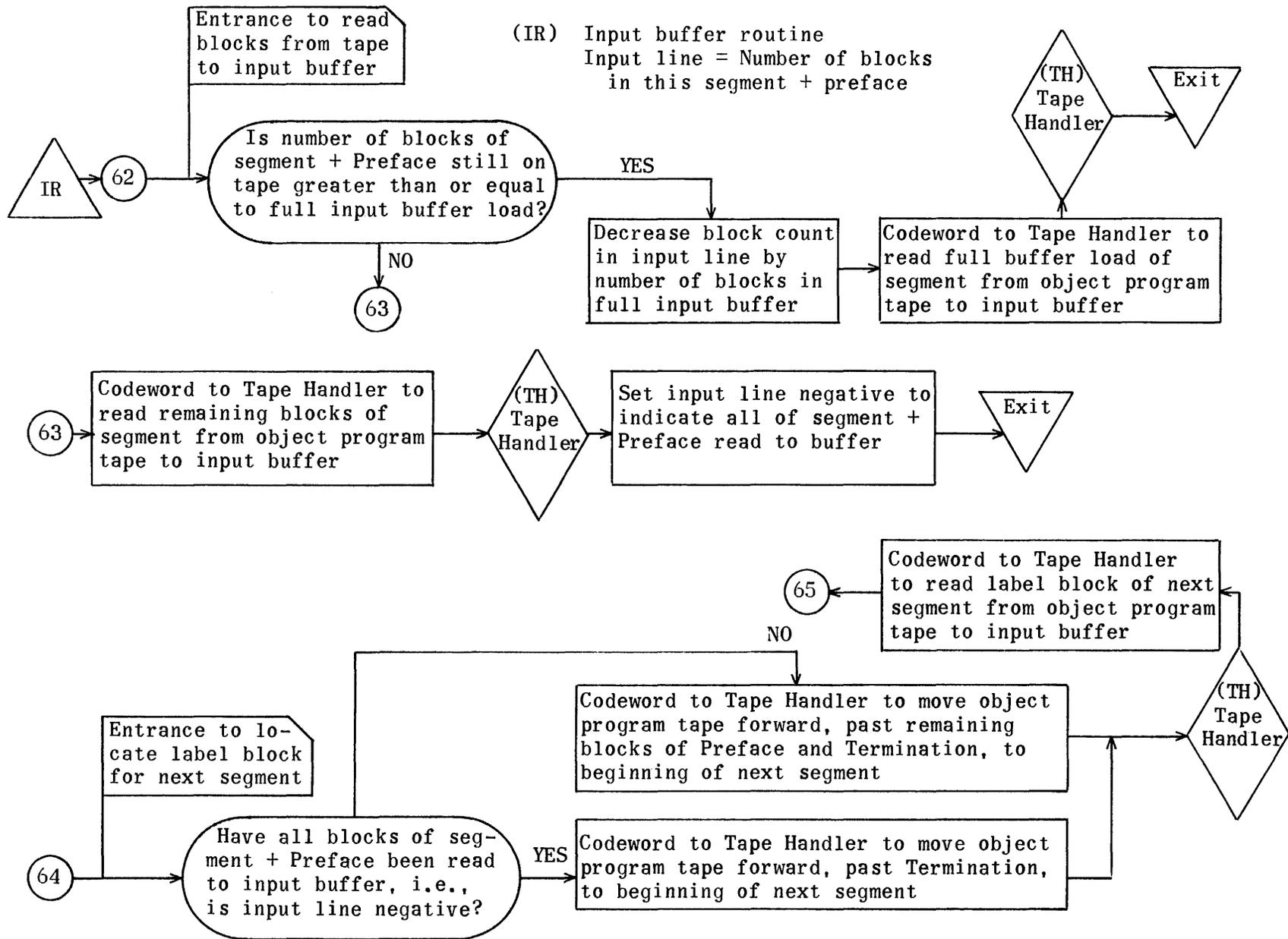
(DS) Dimension List Search Routine

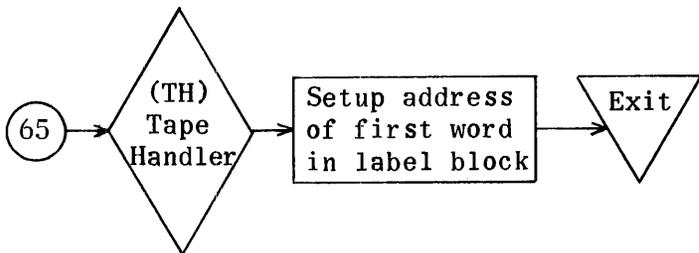




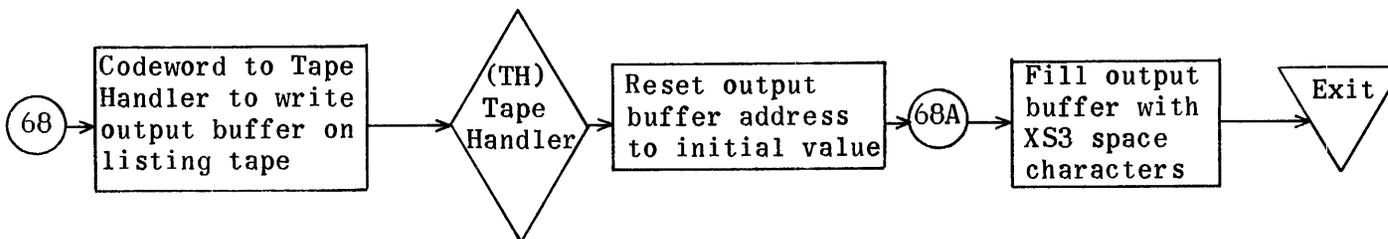
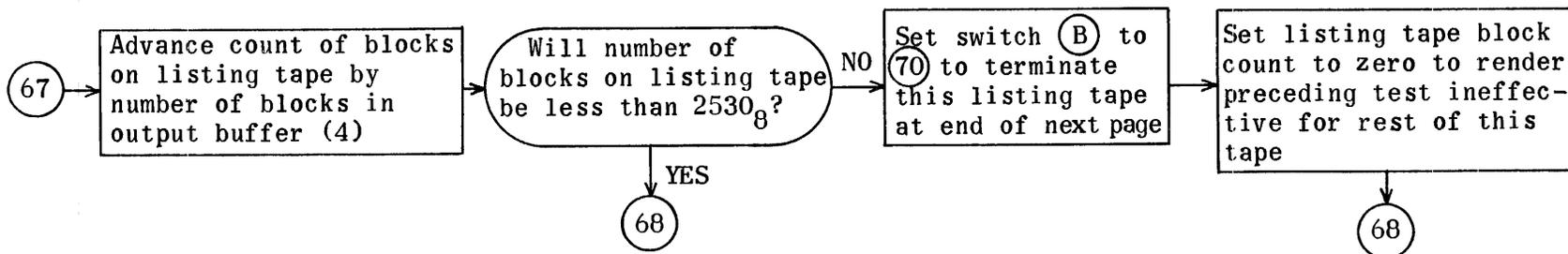
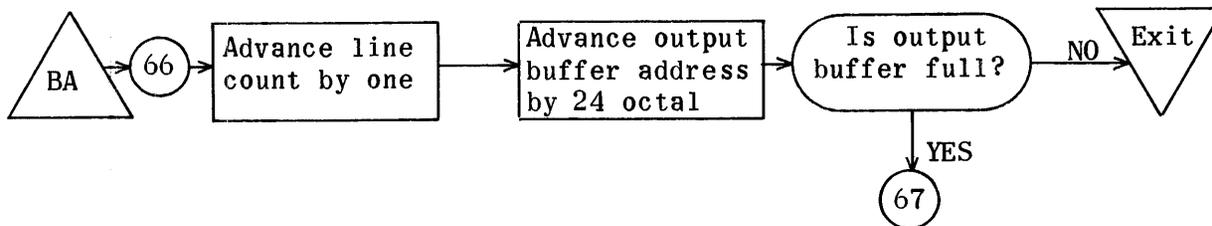
3 repeat tp's in Termination for this array

2 repeat tp's in Termination for this array

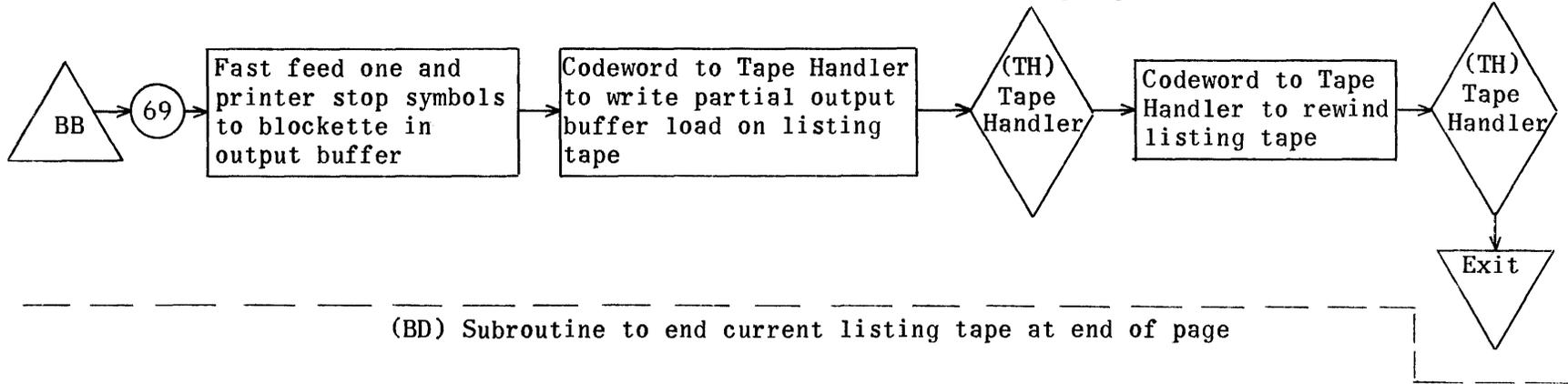




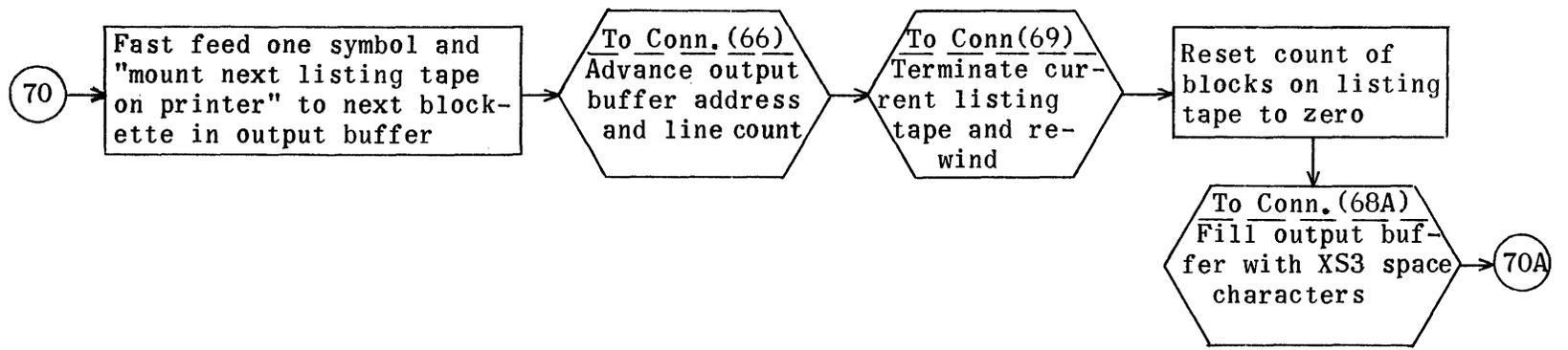
(BA) Subroutine to advance output buffer address



(BB) Subroutine to terminate current listing tape

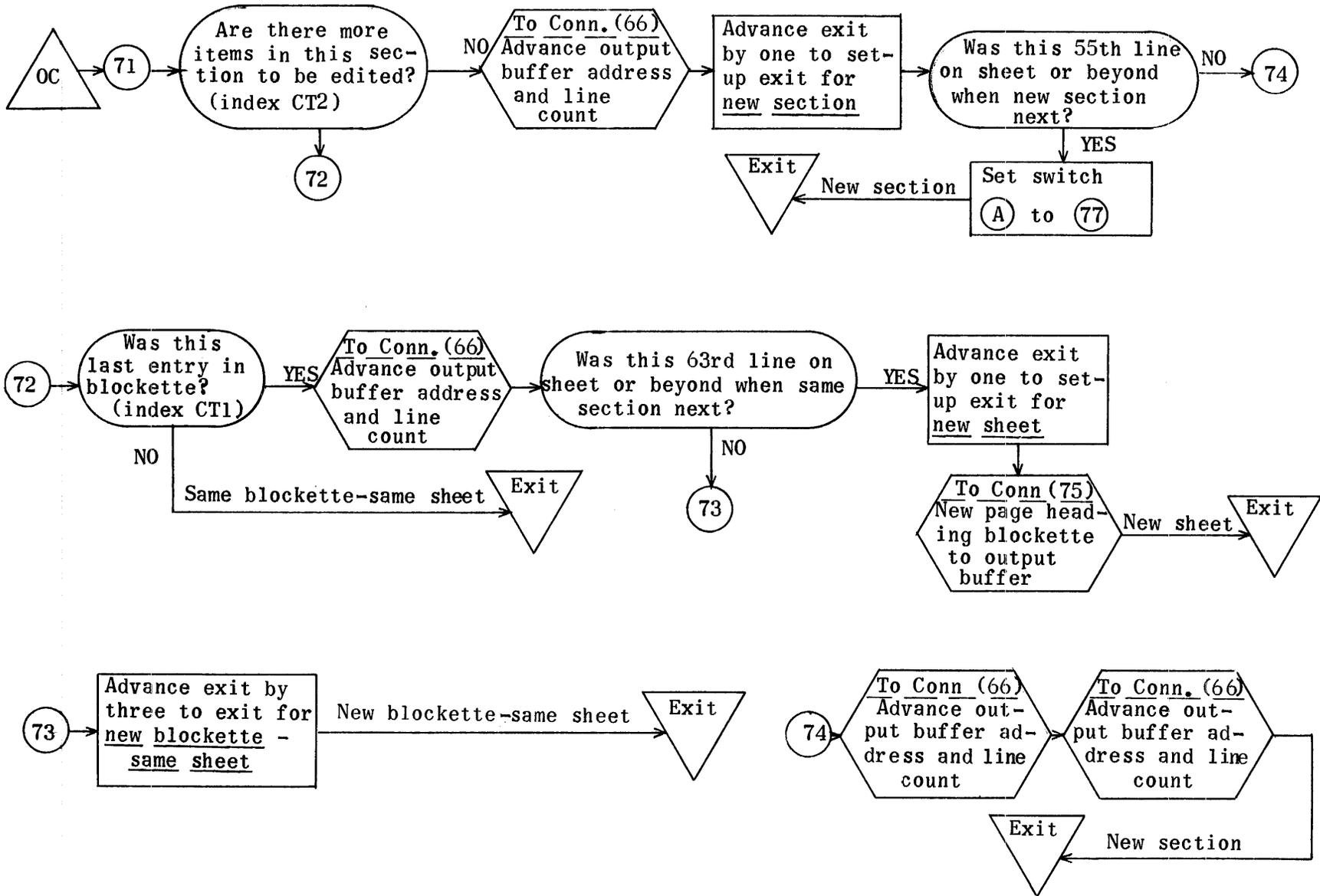


(BD) Subroutine to end current listing tape at end of page

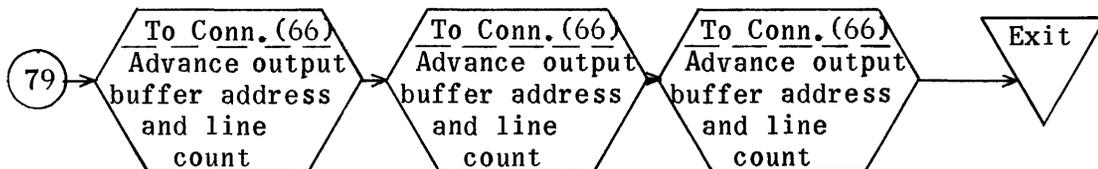
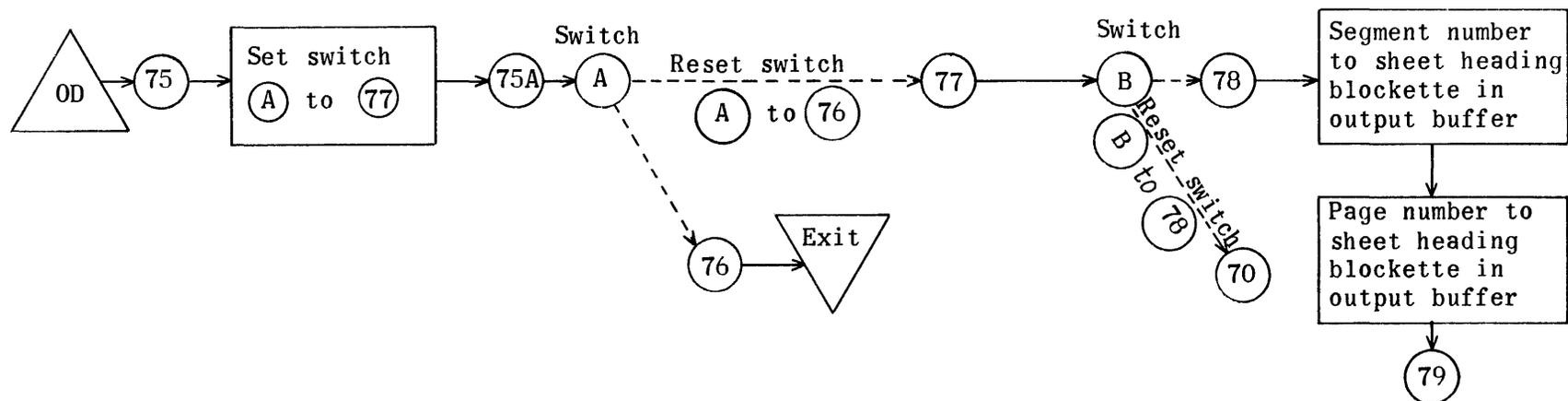


1782

(OC) Output control subroutine

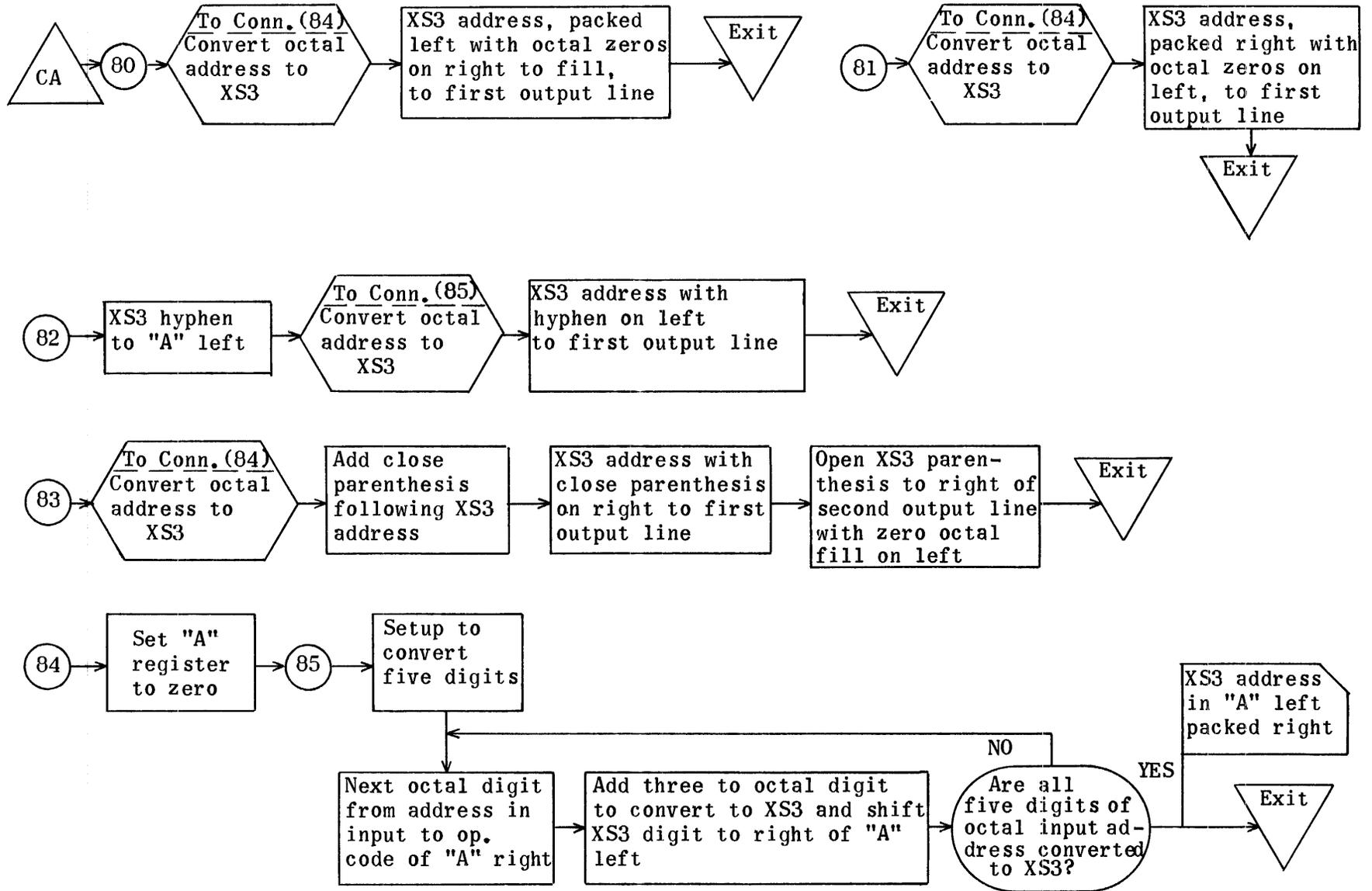


(OD) Page Heading Control Subroutine



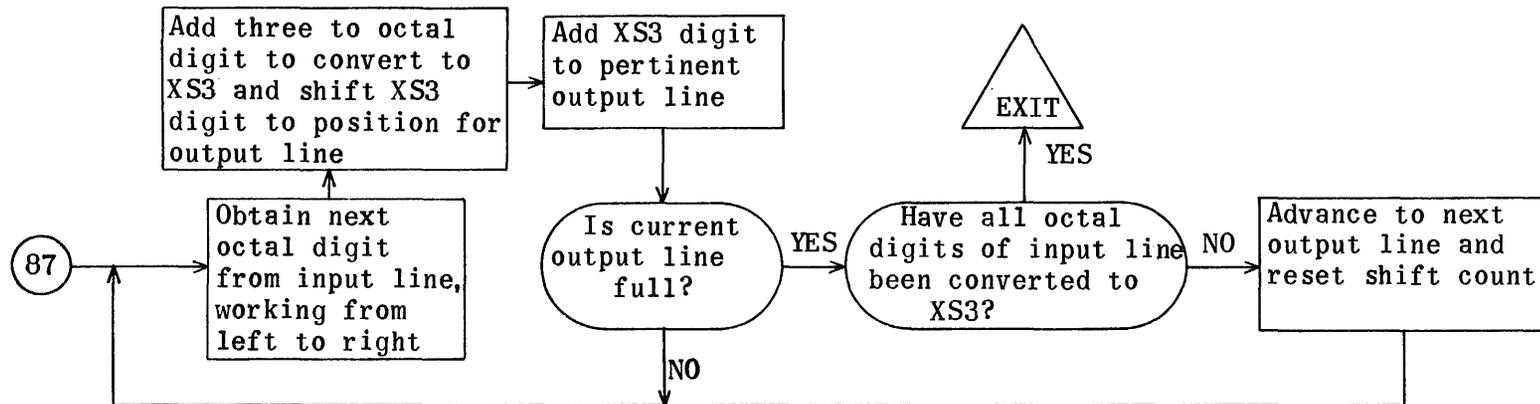
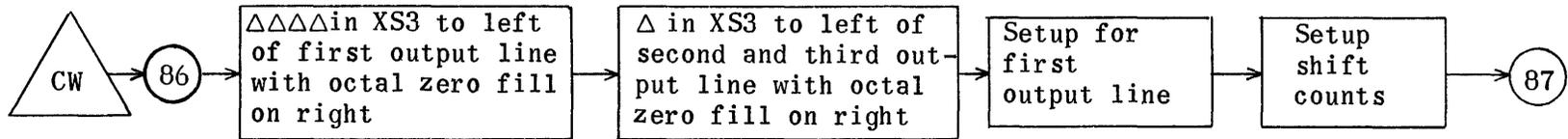
(CA) Convert octal address to XS3

Input = Octal address    Output = XS3 address

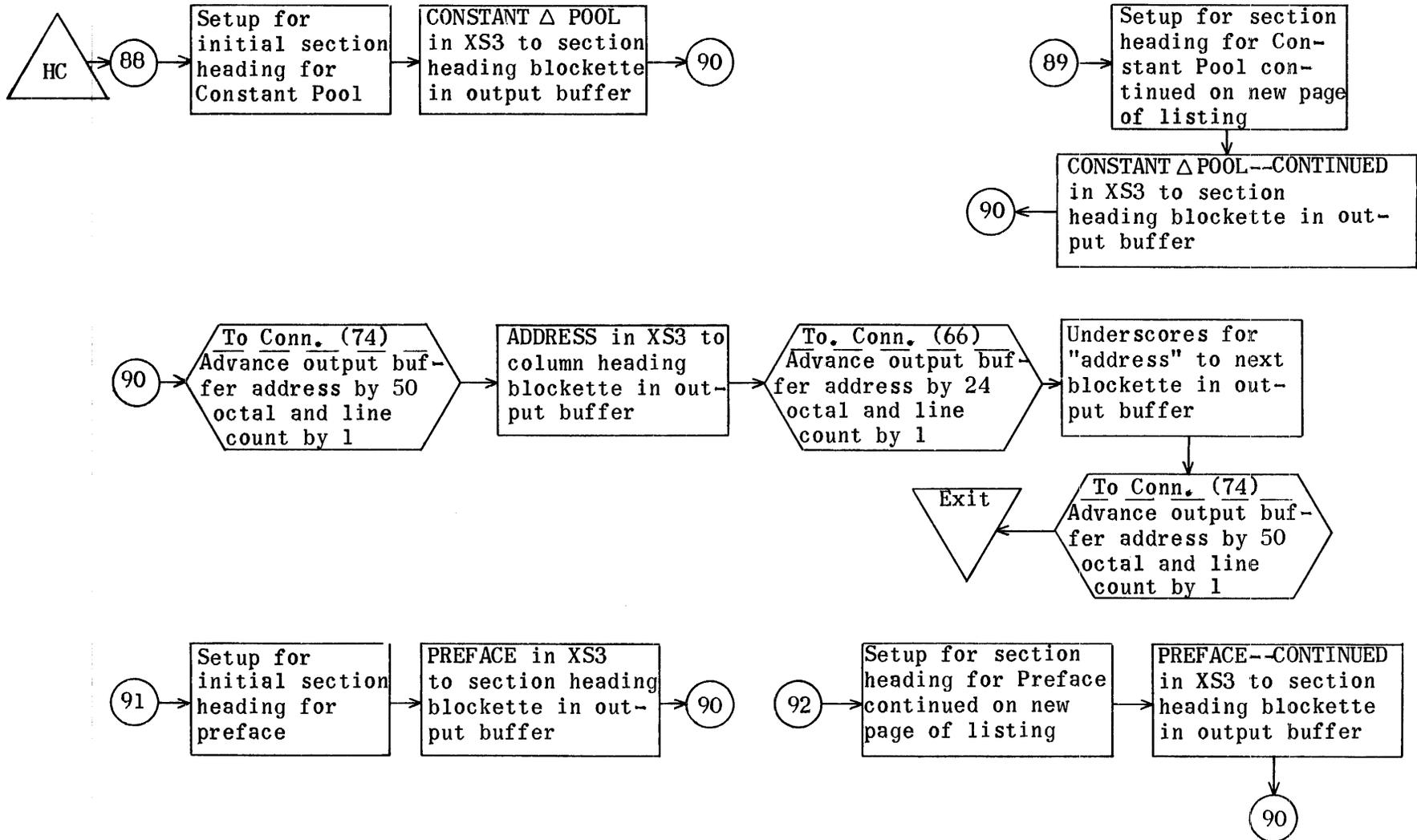


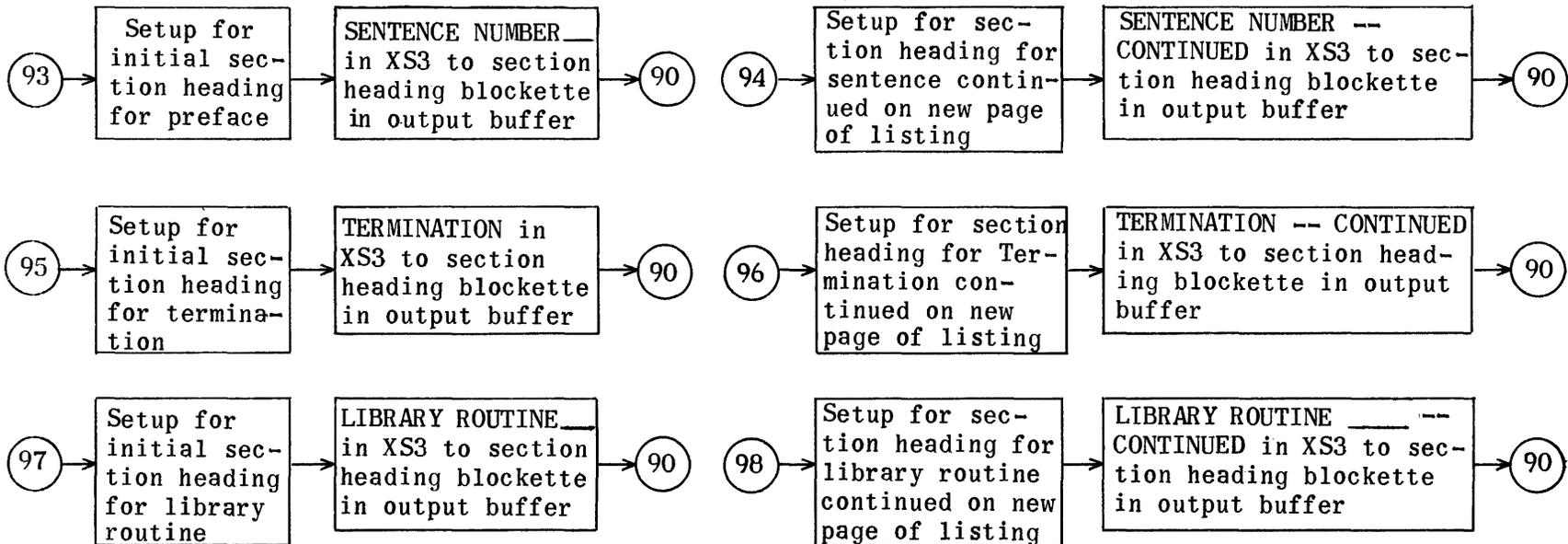
(CW) Convert Octal word to XS3  
 Input = octal computer word

First output line = op. code in XS3  
 Second output line = "u" address in XS3  
 Third output line = "v" address in XS3

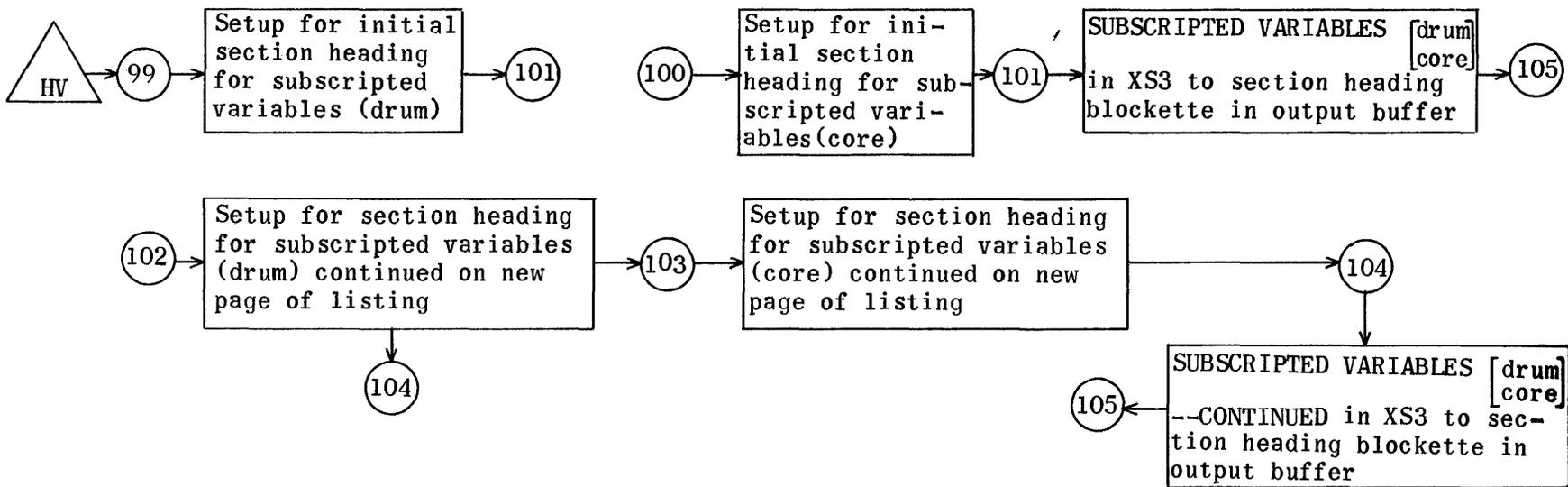


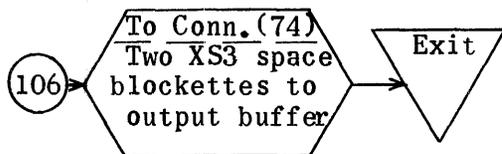
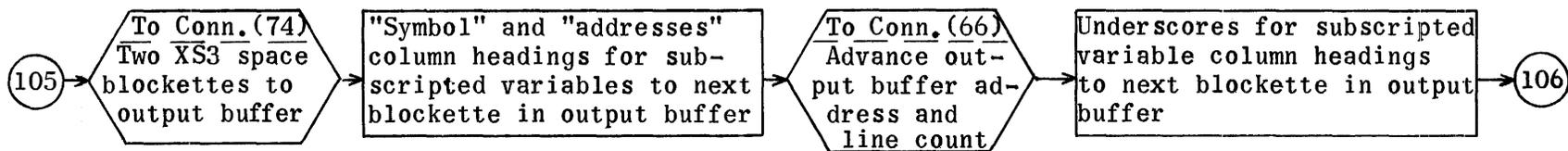
(HC) Heading Routine for Constant Pool, Preface, Sentence, Library Routine, and Termination Sections



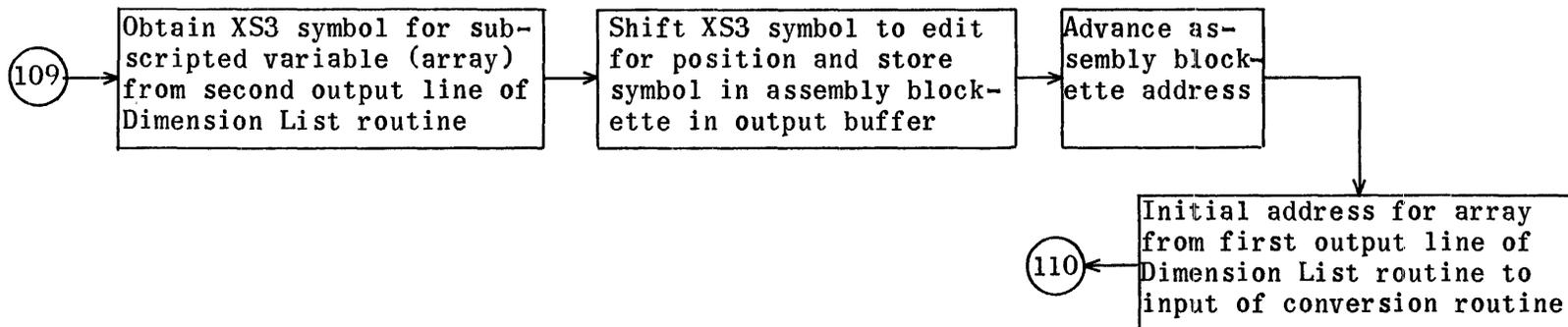
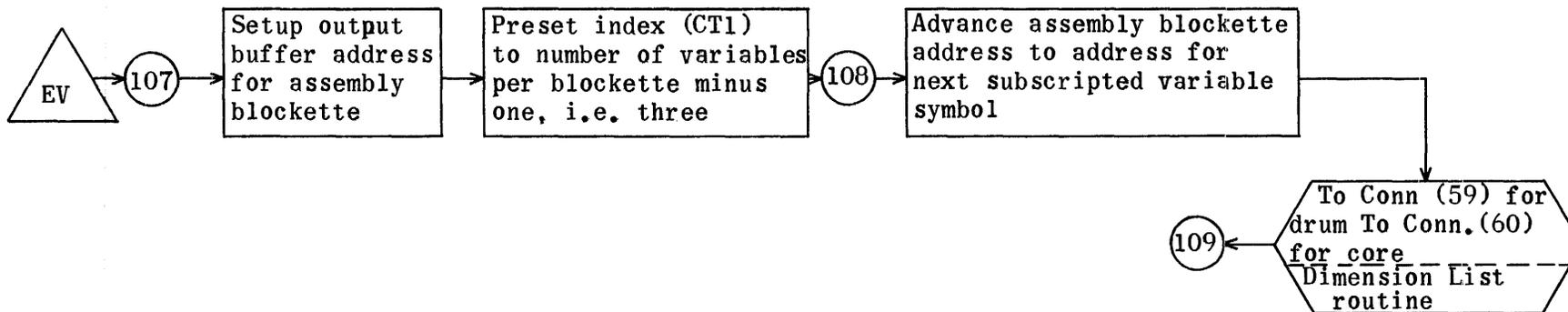


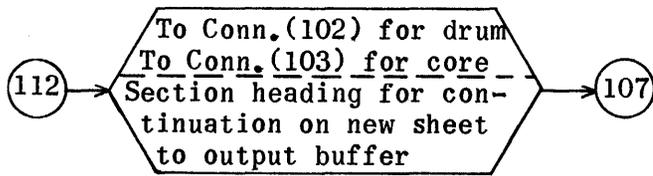
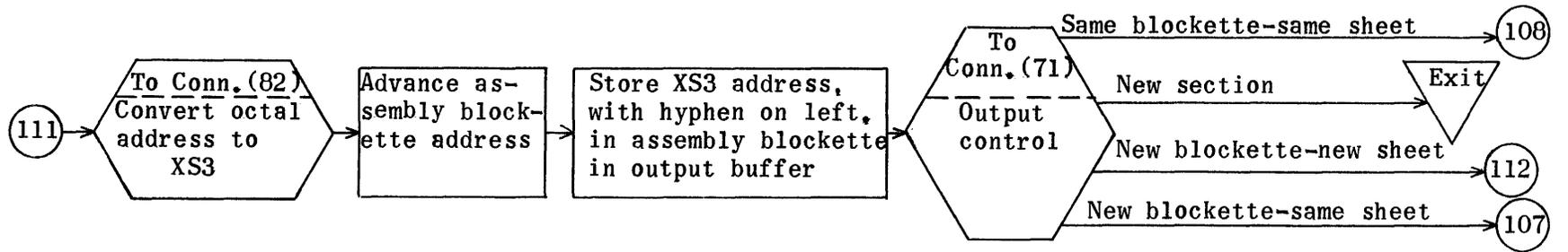
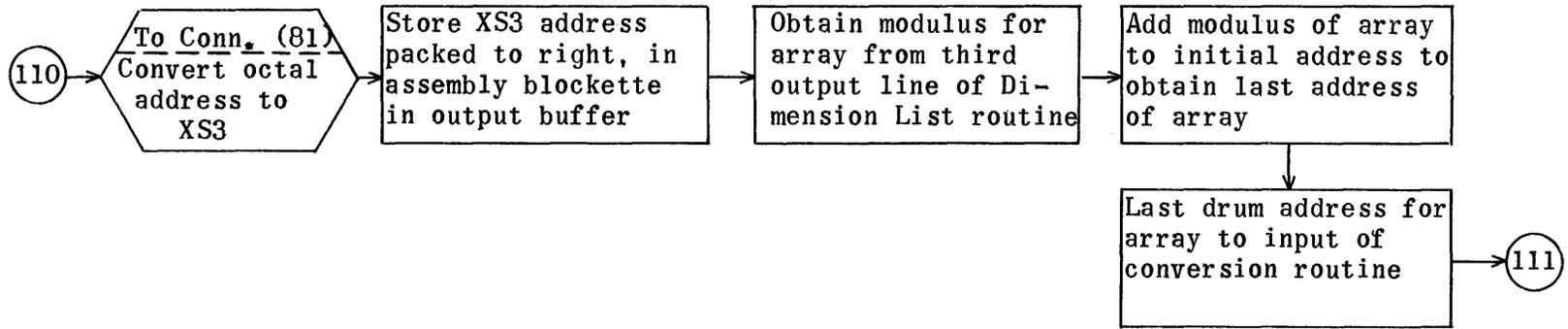
(HV) Heading routine for subscripted variables



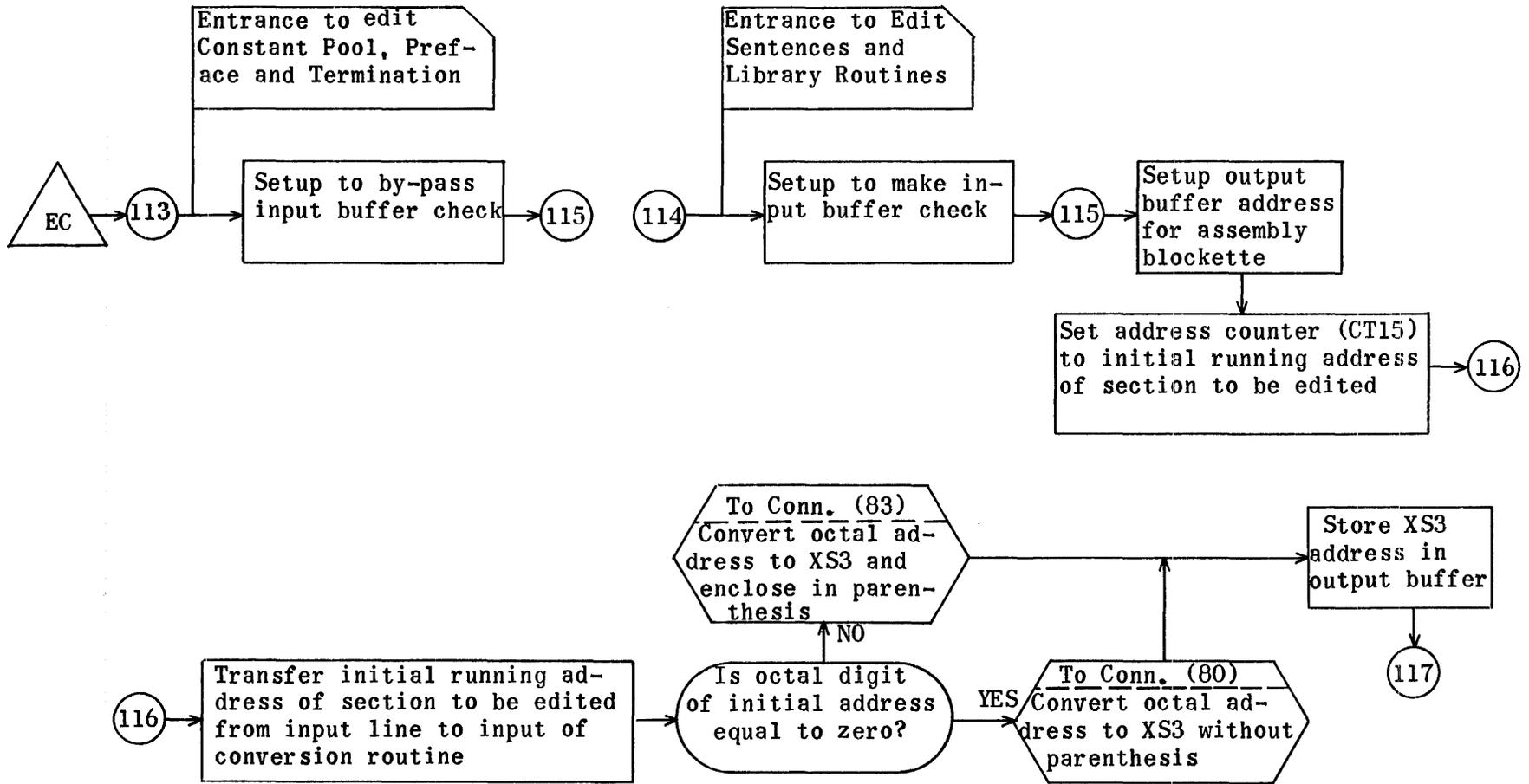


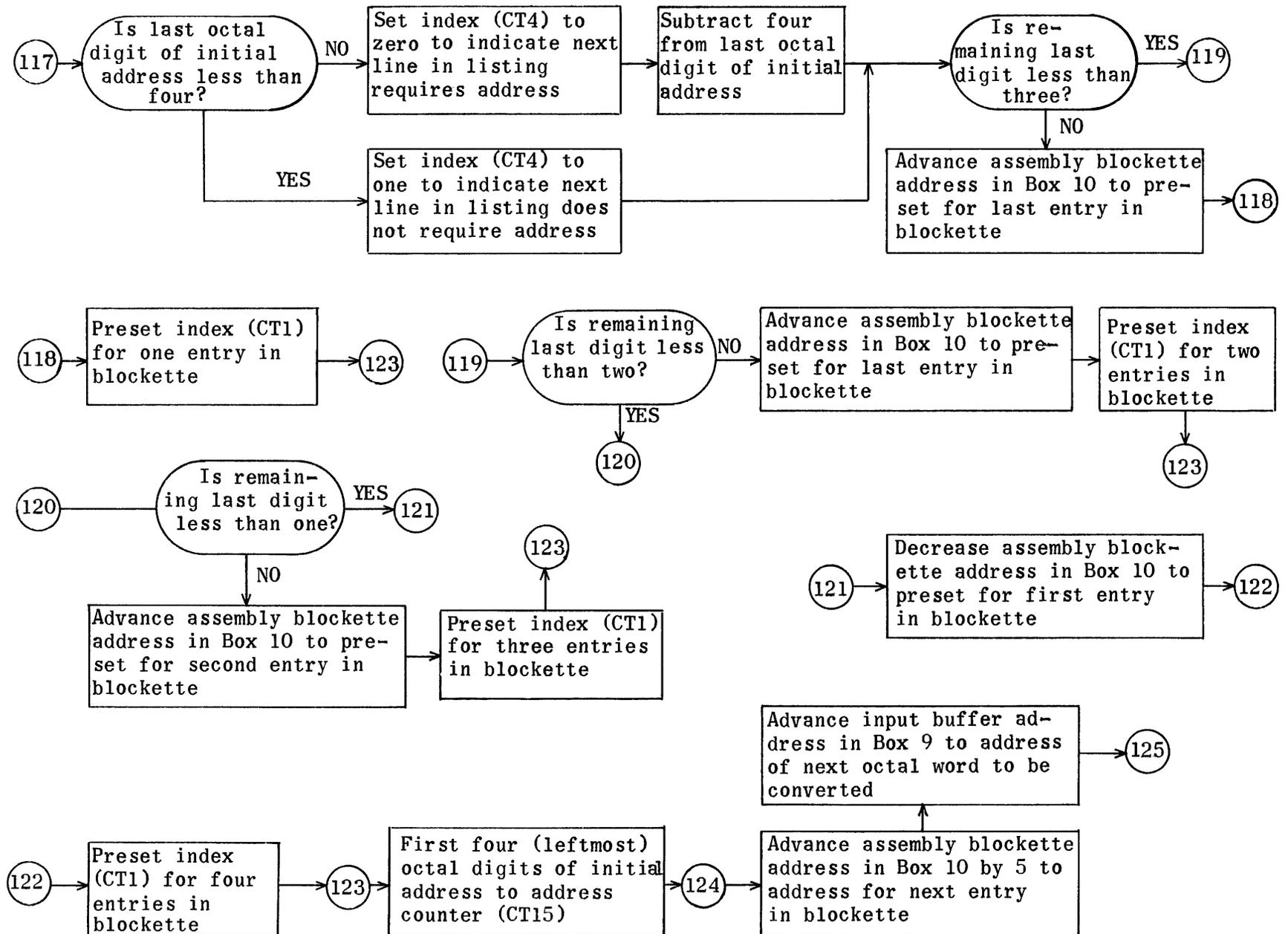
(EV) Edit Subscripted Variables (Drum or Core) and Write on Listing Tape

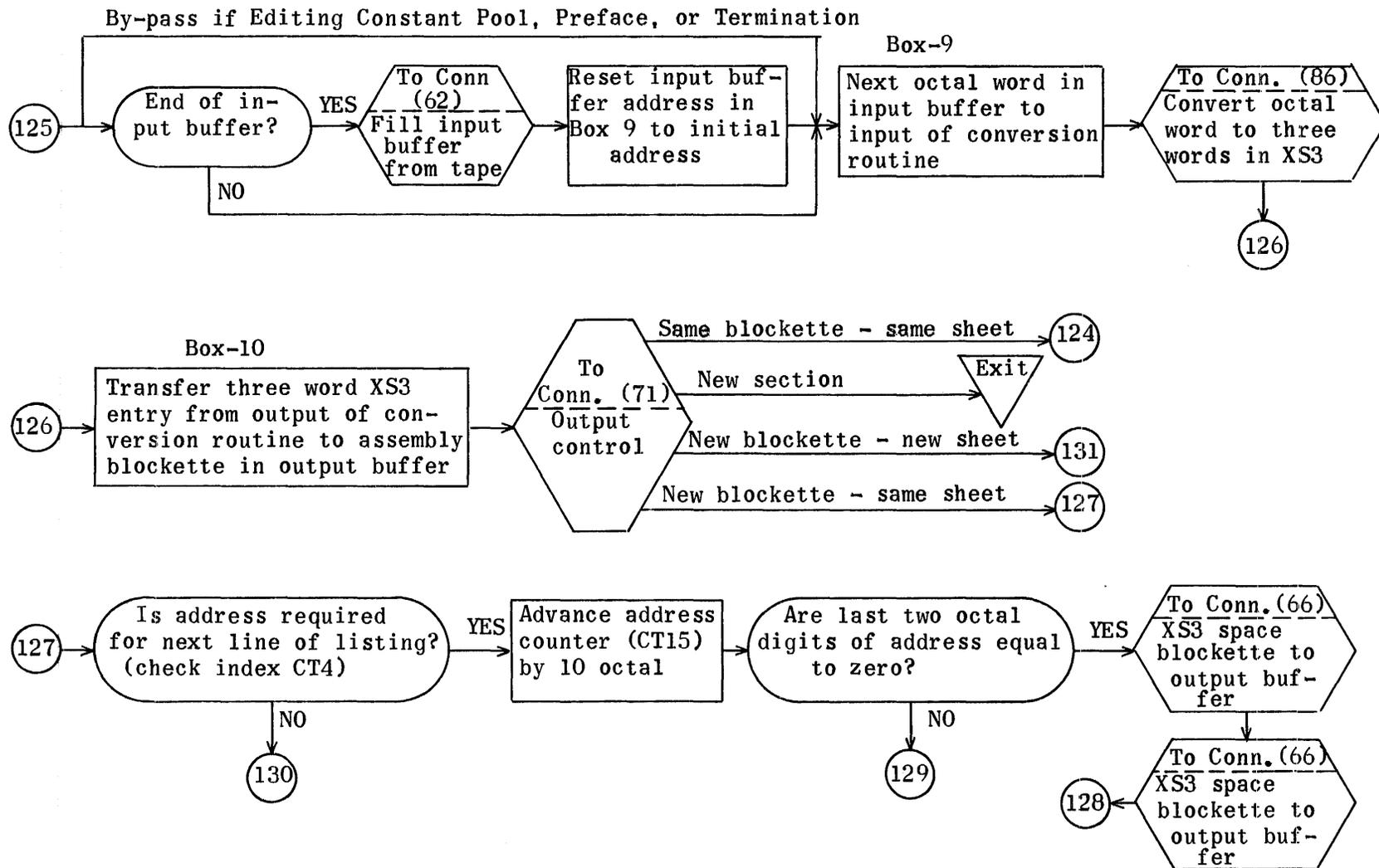


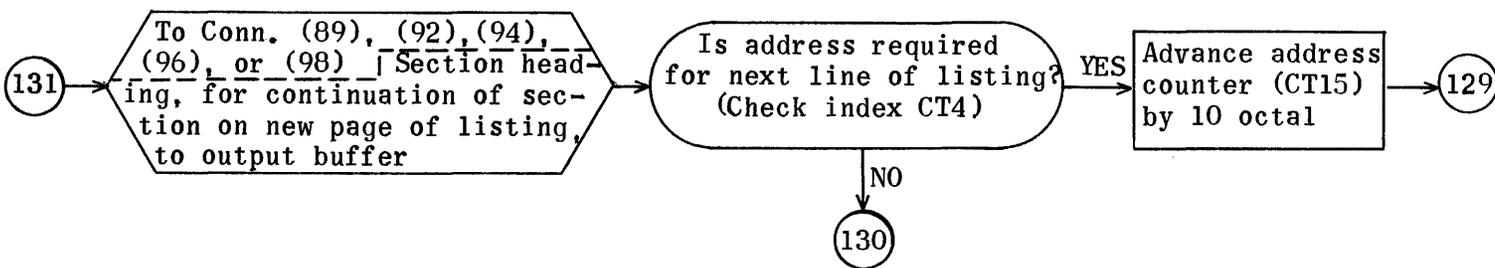
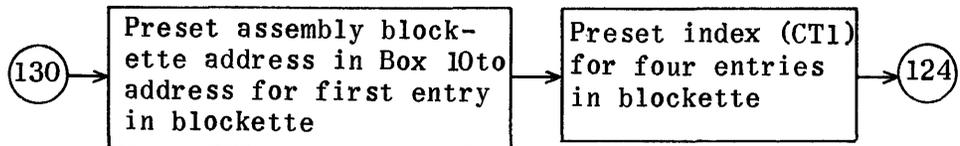
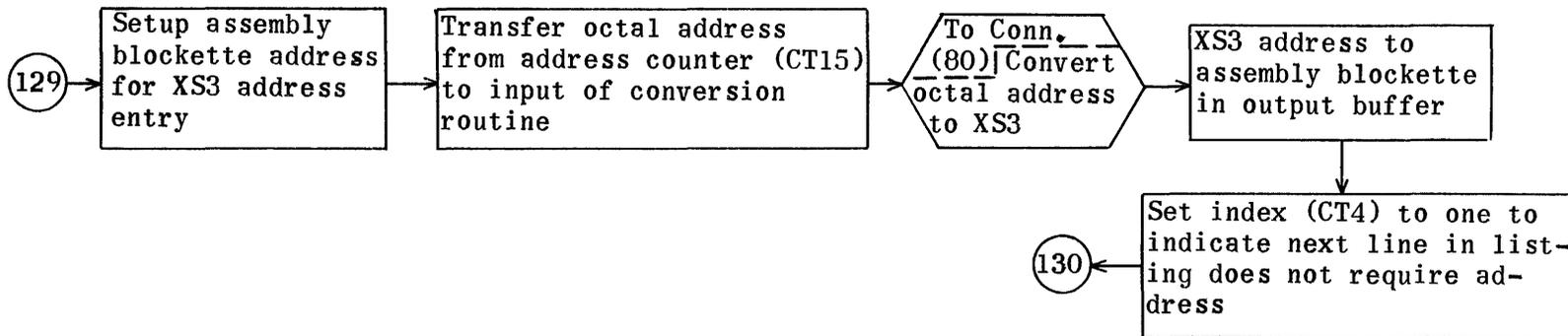
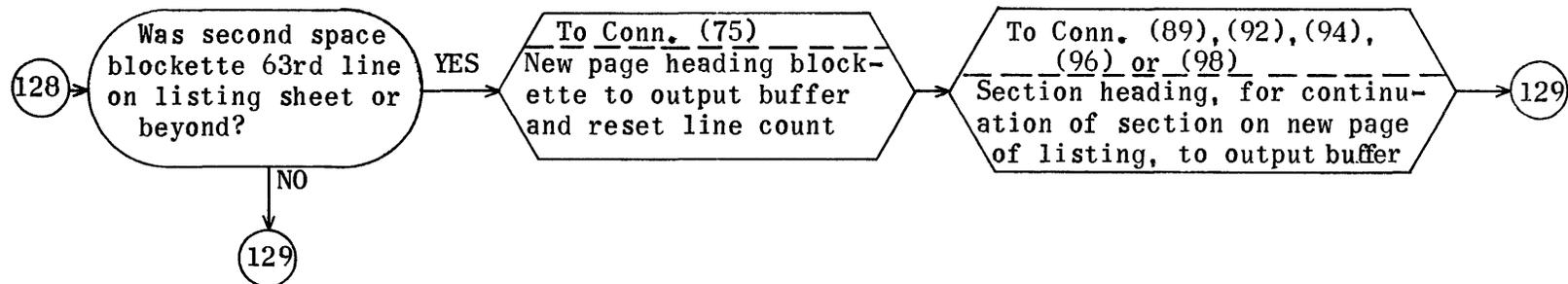


(EC) Edit octal coding or constants and write on listing tape  
 Input - Initial object program running address of section to be edited







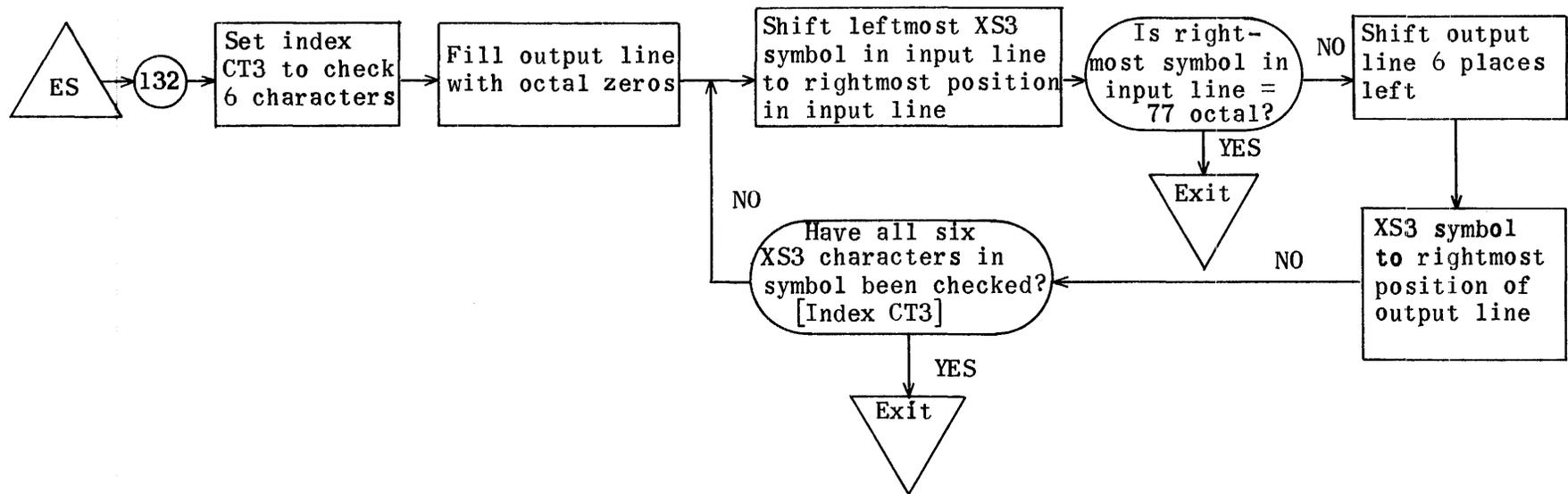


1794

## (ES) Edit XS3 Variable Symbol for Octal 77's

Input = XS3 symbol packed left with octal 77 fill

Output = XS3 symbol packed right with octal zero fill



### Program Listing Phase Regions

	RE	BR537	Alarm routine
	RE	TH21	Tape handler
	RE	UP421	Uniprint routine
	RE	EP540	≡ BR1 Alarm 1
	RE	WP551	≡ BR12 Alarm 10
	RE	WQ551	≡ BR12 Alarm 10
	RE	WV547	≡ BR10 Alarm 8
	RE	YP551	≡ BR12 Alarm 10
	RE	YQ551	≡ BR12 Alarm 10
	RE	YV551	≡ BR12 Alarm 10
	RE	ZP547	≡ BR10 Alarm 8
Group I	RE	FP653	
	RE	TL732	
	RE	TC746	
	RE	XS772	
	RE	XT1072	
	RE	FC1146	
	RE	RC1252	
	RE	CT1316	
	RE	OD1342	
	RE	NP1363	
	RE	BA1415	
	RE	BB1432	
Group II	RE	PP1452	
	RE	PT1467	
	RE	BF1510	
	RE	BG1611	
Group III	RE	PQ1672	
	RE	PR1725	
	RE	PS1770	
	RE	EV2030	
	RE	EC2064	
	RE	ED2125	
	RE	EF2177	
	RE	FR2203	
	RE	IR2211	
	RE	DS2241	
	RE	OC2311	
	RE	BD2335	
	RE	HV2352	
	RE	HC2407	
	RE	CA2470	
	RE	CW2521	
Group IV	RE	PK2547	
	RE	PL2565	
	RE	PM2625	

RE	PN2660	
RE	P02741	
RE	CL2756	
RE	LS2767	
RE	ES3000	
RE	LB3327	List Buffer
RE	DL5037	Modified Dimension list in core
RE	OB7040	Output buffer
RE	FB2170	File buffer
RE	SB2360	Statement buffer
RE	RB2550	Routine buffer
RE	NL2740	Sentence number list in core
RE	FL2557	Op. File IV list in core
RE	IB2747	Input buffer
RE	DD40101	Modified Dimension list on drum
RE	ND42102	Sentence number
RE	FD46202	Op. File IV on drum
RE	RF47202	Routine file for Op. File IV
RE	DQ52472	Group III instructions on drum
RE	ZZ655	Length of Group III
RE	DP53400	Group II instructions on drum
RE	YY220	Length of Group II
RE	TB610	Initial address of termination buffer
RE	BL2260	Listing tape block limit

Memory Layout

TH 21	GP. IV	PK2547	EP 537	≡ BR
400		PL2565	WP 551	≡ BR12
UP 421		PM2625	WQ 551	≡ BR12
216		PN2660	WV 547	≡ BR10
EP 537		PO2741	YP 551	≡ BR12
-----		CL2756	YQ 551	≡ BR12
GP. I		LS2767	YV 551	≡ BR12
FP 653		ES3000	ZP 547	≡ BR10
TL 732				
TC 746				
XS 772				
XT1072	GP. IV	LB3327		
FC1146	(Lists &	1510		
RC1252	buffers)	DL5037		
CT1316		2001		
OD1342		OB7040		
NP1363		740		
BA1415		<u>10000</u>		
BB1432				
-----				
GP. II	GP. II	FB2170		
PP1452	(Lists &	170		
PT1467	buffers)	SB2360		
BF1510		170		
BG1611		RB2550		
		170		
		NL2740		
		4100		
		OB7040		
		<u>740</u>		
		10000		
GP. III	GP. III	FL2557		
PQ1672	(Lists &	170		
PR1725	buffers)	IB2747		
PS1770		2070		
EV2030		DL5037		
EC2064		2001		
ED2125		OB7040		
EF2177		<u>740</u>		
FR2203		10000		
IR2211				
DS2241				
OC2311				
BD2335				
HV2352				
HC2407				
CA2470				
CW2521				
-----				

Not included on region tape since referenced only by uniprint.

FC	40	001	
		<u>100</u>	
DD	40	101	Dimension List
		<u>2 001</u>	
ND	42	102	Sentence number list
		<u>4 100</u>	
FD	46	202	Op. File IV on drum
		<u>1 000</u>	
RF	47	202	Routine File on drum
		<u>3 270</u>	
DQ	52	472	Group III on drum
		<u>ZZ 655</u>	Length Group III
DP	53	400	Group II on drum
YY		<u>220</u>	Length Group II
TB	610		Termination buffer address
BL	2260		Listing tape block limit

Program Listing Phase

0	IA	PK		Program listing setup
1	TP	FP	UP3	Parameter → uniprint
2	RJ	UP2	UP	Print: LISTING OF PROGRAM
3	RP	10024	PK4	
4	TP	FC	CT	Zeroize temporaries
5	TP	XS11	CT10	Preset 1st page no. word (assume no number 1st page)
6	TP	XS12	CT11	Preset 2nd page no. word
7	TP	XS	CT12	Preset 1st segment no. word
10	TP	XS11	CT13	Preset 2nd segment no. word
11	TV	RC24	NP4	Preset one shot jump in Page no. rtn.
12	TV	RC25	NP7	Preset one shot jump in Page no. rtn.
13	RP	YY30000	PK14	} Program Load II → drum
14	TP	PP	DP	
15	RP	ZZ30000	PL	} Program Load III → drum
16	TP	PQ	DQ	
	CA	PK16		

Program Listing (Title Section)

①	0	IA	PL		
	1	RP	10740	PL2	} Fill output buffer with space char. Fast feed 1 sym → sheet hdg. blkt.
	2	TP	XS11	OB	
	3	TP	XS	OB	
	4	RP	30004	PL5	
	5	TP	XT40	OB10	PROGRAM Δ LISTING → Sheet Hdg. blkt.
	6	TP	TC	TH3	Codeword → G.T.H.
	7	RJ	TH2	TH	Read 1 blk. corrected problem tape → list buffer
	8	RJ	CL	CL1	Check corr. prob. tape label (i.e. UNICOD <del>E</del> ΔPROGRAM)
②	10	RP	30144	PL12	
	11	TP	LB24	OB50	Prog. title → 3rd - 7th blks in output buffer
	12	TP	RC3	CT6	Preset output buffer address
	13	TP	FC7	CT7	Preset line count (15 ₈ ) for 1st entry following title
	14	SP	14	0	# blks preceding XS3 sym. list lab → Av
	15	ST	FC1	Q	Decrease by 1 to exclude tape label blk. → Q _v
	16	LT	3	CT16	# blks. Const. Pool (incl. lab. blk. & End blk.) → "v" of temp.
	17	QT	FC32	A	# blks. to move tape to position at begin XS3 sym list lab.
	20	SS	CT16	25	Dec. by # blks. Const. Pool to get # blks. to move to begin const. pool
	21	AT	TC3	TH3	Codeword → G.T.H.
	22	RJ	TH2	TH	Move corr. prob. tape forward to begin Const. pool (or XS3 sym. list if no C.P.)
	23	MJ	10000	PL30	MJ1 off ⇒ 5 servos; MJ1 on ⇒ 7 servos
	24	TP	TC21	Q	Obj. prog. servo # = 3 → Q (5 servos)
	25	TP	FP55	FP20	Set listing tape # = 4 in flex. prints
	26	TP	FP55	FP47	Set listing tape # = 4 in flex. prints
	27	MJ	0	PL33	
	30	TP	FP56	FP20	Set listing tape # = 7 in flex. prints
	31	TP	FP56	FP47	Set listing tape # = 7 in flex. prints
	32	TP	TC22	Q	Obj. prog. servo # = 6 → Q (7 servos)
	33	RP	30010	PL35	

34	QT	TC6	TC6	Servo. no. → Obj. prog. tape codeword
35	RA	Q	FC2	Program listing servo no. → Q (1 in "u" adv.)
36	RP	30003	PM	
37	QT	TC16	TC16	Servo no. → program listing tape codewords
	CA	PL40		

Listing Phase (Subscripted Var. (Drum) Section)

③	0	IA	PM						
	1	TV	6	CT2					# 77--- CW's → Index C ₂
	2	IJ	CT2	PM3	yes				Are there subscripted variables?
	3	MJ	0	PM15					No
	4	TP	6	A					jn for Dim. List of form 2---- → Au
	5	AT	FC77	5					jn for Dim. List of form 3----
	6	TU	A	PM6					→ "u" of loc. 5
	7	RP	[30000]	PM10					Dimension list from drum → core
	10	TP	DD	DL					
	11	TU	RC33	DS4					
	12	TV	RC42	EV7					Preset init. add. dim. list
	13	TV	RC5	EV32					Preset Dim. List rtn. ref. → subs.
	14	RJ	HV	HV1					var. (drum) entry
	15	RJ	EV	EV1					Preset hdg. rtn. ref. → subs. var.
	16	RJ	EV	EV1					(drum W/cont.) entry
	17	RJ	EV	EV1					Init. subs. var. (drum) hdgs.
	18	RJ	EV	EV1					→ sect. hdg. blk.
	19	RJ	EV	EV1					Edit subs. var. (drum) & write on
④	20	SP	CT16	25					listing tape
	21	SP	CT16	25					# blks const. pool (incl. lab. &
	22	ZJ	PM17	yes	PN	no			end blks.) → A in codeword posi-
	23	AT	TC5		TH3				tion
	24	RJ	TH2		TH				Is there const. pool?
	25	RJ	TH2		TH				Codeword → G.T.H.
	26	RJ	TH2		TH				Read const. pool (incl. lab. & end)
	27	RJ	TH2		TH				from corr. prob. tape → Dim. List
	28	RJ	TH2		TH				region
	29	RJ	TH2		TH				1st word const. pool lab. blk.
	30	RJ	TH2		TH				→ A
	31	RJ	TH2		TH				1st word const. pool lab. blk. =
	32	RJ	TH2		TH				C O N S T A ?
	33	RJ	TH2		TH				Alarm 10
	34	RJ	TH2		TH				# blks XS3 sym. list incl. lab. &
	35	RJ	TH2		TH				end blks. (If no sym. list, only
	36	RJ	TH2		TH				Lab. blk. appears & count = 1)
	37	RJ	TH2		TH				→ A in position for codeword
	38	RJ	TH2		TH				Codeword to read sym. list to
	39	RJ	TH2		TH				list buffer → G.T.H.
	40	RJ	TH2		TH				Read XS3 sym. list (or lab. blk.
	41	RJ	TH2		TH				if no list) → List buffer
	42	RJ	TH2		TH				1st word XS3 sym. list lab.
	43	RJ	TH2		TH				blk → A
	44	RJ	TH2		TH				1st word XS3 sym. list lab.
	45	RJ	TH2		TH				blk = S Y M B O L ?
	46	RJ	TH2		TH				Alarm 10
	47	RJ	TH2		TH				
	48	RJ	TH2		TH				
	49	RJ	TH2		TH				
	50	RJ	TH2		TH				
	51	RJ	TH2		TH				
	52	RJ	TH2		TH				
	53	RJ	TH2		TH				
	54	RJ	TH2		TH				
	55	RJ	TH2		TH				
	56	RJ	TH2		TH				
	57	RJ	TH2		TH				
	58	RJ	TH2		TH				
	59	RJ	TH2		TH				
	60	RJ	TH2		TH				
	61	RJ	TH2		TH				
	62	RJ	TH2		TH				
	63	RJ	TH2		TH				
	64	RJ	TH2		TH				
	65	RJ	TH2		TH				
	66	RJ	TH2		TH				
	67	RJ	TH2		TH				
	68	RJ	TH2		TH				
	69	RJ	TH2		TH				
	70	RJ	TH2		TH				
	71	RJ	TH2		TH				
	72	RJ	TH2		TH				
	73	RJ	TH2		TH				
	74	RJ	TH2		TH				
	75	RJ	TH2		TH				
	76	RJ	TH2		TH				
	77	RJ	TH2		TH				
	78	RJ	TH2		TH				
	79	RJ	TH2		TH				
	80	RJ	TH2		TH				
	81	RJ	TH2		TH				
	82	RJ	TH2		TH				
	83	RJ	TH2		TH				
	84	RJ	TH2		TH				
	85	RJ	TH2		TH				
	86	RJ	TH2		TH				
	87	RJ	TH2		TH				
	88	RJ	TH2		TH				
	89	RJ	TH2		TH				
	90	RJ	TH2		TH				
	91	RJ	TH2		TH				
	92	RJ	TH2		TH				
	93	RJ	TH2		TH				
	94	RJ	TH2		TH				
	95	RJ	TH2		TH				
	96	RJ	TH2		TH				
	97	RJ	TH2		TH				
	98	RJ	TH2		TH				
	99	RJ	TH2		TH				
	100	RJ	TH2		TH				
	101	RJ	TH2		TH				
	102	RJ	TH2		TH				
	103	RJ	TH2		TH				
	104	RJ	TH2		TH				
	105	RJ	TH2		TH				
	106	RJ	TH2		TH				
	107	RJ	TH2		TH				
	108	RJ	TH2		TH				
	109	RJ	TH2		TH				
	110	RJ	TH2		TH				
	111	RJ	TH2		TH				
	112	RJ	TH2		TH				
	113	RJ	TH2		TH				
	114	RJ	TH2		TH				
	115	RJ	TH2		TH				
	116	RJ	TH2		TH				
	117	RJ	TH2		TH				
	118	RJ	TH2		TH				
	119	RJ	TH2		TH				
	120	RJ	TH2		TH				
	121	RJ	TH2		TH				
	122	RJ	TH2		TH				
	123	RJ	TH2		TH				
	124	RJ	TH2		TH				
	125	RJ	TH2		TH				
	126	RJ	TH2		TH				
	127	RJ	TH2		TH				
	128	RJ	TH2		TH				
	129	RJ	TH2		TH				
	130	RJ	TH2		TH				
	131	RJ	TH2		TH				
	132	RJ	TH2		TH				
	133	RJ	TH2		TH				
	134	RJ	TH2		TH				
	135	RJ	TH2		TH				
	136	RJ	TH2		TH				
	137	RJ	TH2		TH				
	138	RJ	TH2		TH				
	139	RJ	TH2		TH				
	140	RJ	TH2		TH				
	141	RJ	TH2		TH				
	142	RJ	TH2		TH				
	143	RJ	TH2		TH				
	144	RJ	TH2		TH				
	145	RJ	TH2		TH				
	146	RJ	TH2		TH				
	147	RJ	TH2		TH				
	148	RJ	TH2		TH				
	149	RJ	TH2		TH				
	150	RJ	TH2		TH				
	151	RJ	TH2		TH				
	152	RJ	TH2		TH				
	153	RJ	TH2		TH				
	154	RJ	TH2		TH				
	155	RJ	TH2		TH				
	156	RJ	TH2		TH				
	157	RJ	TH2		TH				
	158	RJ	TH2		TH				
	159	RJ	TH2		TH				
	160	RJ	TH2		TH				
	161	RJ	TH2		TH				
	162	RJ	TH2		TH				
	163	RJ	TH2		TH				
	164	RJ	TH2		TH				
	165	RJ	TH2		TH				
	166	RJ	TH2		TH				
	167	RJ	TH2		TH				
	168	RJ	TH2		TH				
	169	RJ	TH2		TH				
	170	RJ	TH2		TH				
	171	RJ	TH2		TH				
	172	RJ	TH2		TH				
	173	RJ	TH2		TH				
	174	RJ	TH2		TH				
	175	RJ	TH2		TH				
	176	RJ	TH2		TH				
	177	RJ	TH2		TH				
	178	RJ	TH2						

Non-Subscripted Variable Section

⑤	0	IA SP	PN 7	0	Init. running add. non-subs. var. → Au, # non-subs. var. → Av			
	1	ST	FC3	CT	Decrease running add. & # non-subs. var. each by 1 → temp			
	2	SJ	PO ^{no}	PM24 ^{yes}	Are there non-subs. var.? No ⇒ const. pool section			
⑥	3	TV	CT	CT2	# non-subs. var. - 1 → index C ₂			
	4	TV	FC	CT	Zero → "v" of temp. containing non-subs. var. add. - 1			
	5	RJ	OD	OD2	New page hdg. if required → out- put buffer			
	6	TU	RC12	PN37	Preset initial add. in XS3 sym. list			
	7	TP	XS53	XS50	Setup section hdg.			
	10	TV	CT6	PN12	Preset add. sect. hdg. blkt = output buffer add.			
	11	RP	30005	PN13				
	12	TP	XS44	[30000]	"Non-subscripted variables" Hdg. → Section hdg. blkt.			
	⑥A	13	TU	RC10	PN21	Preset "u" of TP → Add. of stored col. hdgs.		
		14	RJ	OC	OC21	Two space blkts. → output buffer		
⑦	15	TP	FC20	Q	Switch <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>06</td><td>00000</td><td>00000</td></tr></table> → Q	06	00000	00000
	06	00000	00000					
	16	TV	CT6	PN21	Preset add. col. hdg. blkt. = out- put buffer add.			
17	RA	PN21	FC10	2 in "v" adv. → add. for 1st col. hdg. (or underscore)				
20	RP	30003	PN22					
21	TP	[30000]	[30000]	Column hdg. (or underscores) → output buffer				
22	RA	PN21	FC21	5 in "v" adv. → Add. for next column hdg.				
23	QJ	PN24 ^{yes}	PN20 ^{no}	All column hdg. (or underscores) → output buffer?				
24	QJ	PN25 ^{no}	PN31 ^{yes}	Underscores transferred yet?				
25	RJ	BA	BA1	Adv. output buff. add. by 20 ₁₀ (24 ₈ )				
26	TU	RC11	PN21	Preset "u" of TP → Add. stored underscores				
27	TP	FC22	Q	Switch <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>04</td><td>00000</td><td>00000</td></tr></table> → Q	04	00000	00000	
04	00000	00000						
30	MJ	0	PN16					
31	RJ	OC	OC21	Two space blkts → output buffer				
⑧	32	TV	CT6	PN41	Set assem. blkt. add. = output buffer add.			
	33	RS	PN41	FC1	Dec. assem. blkt. add. by 1 → pre- set for 1st var. sym.			
	34	TP	FC11	CT1	Preset index C ₁ → # variables/ blkt. - 1			

⑨	35	RA	CT	FC2	1 in "u" adv. → running add. next var.
	36	RA	PN41	FC11	Adv. assem. blkt. add. by 3 → add. next sym.
	37	TP	[30000]	ES2	XS3 var. sym. packed left W/77 ₈ fill → input edit rtn.
		CA	PN40		
		IA	PN40		
	40	RJ	ES	ES3	Pack symbol to right with zero ₈ fill
	41	TP	ES1	[30000]	XS3 var. symbol packed right → output buffer
⑩	42	RA	PN37	FC2	1 in "u" advance to address next var. symbol
	43	RA	PN41	FC10	Adv. add. assem. blkt. by 2 in "v" → add. for next add. entry
	44	TV	A	PN47	Preset address for variable address entry
	45	TP	CT	CA2	Running add. for next var. → conversion routine
	46	RJ	CA	CA3	Convert octal add. → XS3 W/zero ₈ on right
	47	TP	CA1	[30000]	Running address for variable → output buffer
⑪	50	RJ	OC	OC1	
	51	MJ	0	PN35	⇒ same blockette - same sheet
	52	MJ	0	PO	⇒ new section
	53	MJ	0	PN55	⇒ new blockette - new sheet
	54	MJ	0	PN32	⇒ new blockette - same sheet
⑫	55	TP	XS54	XS50	Set up section hdg. to continue on new sheet
	56	TV	CT6	PN60	Preset address section hdg. blkt. = output buffer add.
	57	RP	30007	PN13	
	60	TP	XS44	[30000]	Non-subscripted $\Delta$ variables -- con- tinued → sect. hdg. blkt.
		CA	PN61		

### Constant Pool Section

	IA	PO			
⑬	0	SP	10	17	Init. running add. const. pool → "u" of A
	1	TU	A	EC1	Init. running add. const. pool → input edit routine
	2	LT	6	Q	jn for constant pool → Qv
	3	QT	FC13	CT2	# const. in const. pool → index C2
	4	IJ	CT2	P06	yes Is there const. pool?
	5	MJ	0	P013	
	6	RJ	OD	OD2	New page hdg. if required → out- put buffer
⑭	7	TU	RC22	ED14	Preset input buff. add. for 1st const. - 1
	10	RJ	HC	HC1	Constant pool hdgs. → output buf- fer
	11	TV	RC33	EF	Preset ent. add. for const. pool hdgs. (W/cont.)
	12	RJ	EC	EC2	Edit const. pool & write on list- ing tape
⑮	13	RJ	LS	LS1	Locate 1st segment label blk.
	14	MJ	0	PP	
		CA	P015		

Segment Section

	IA	PP			
①⑥	0	TU	CT	PP1	Address seg. lab. blk. → "u" of NI
	1	TP	[30000]	A	1st word label blk. → A
	2	EJ	TL	PT yes	1st word label blk. = Z's? i.e. is this end obj. prog.?
	3	EJ	TL1	PP5	1st word label blk. = SEGMEN?
	4	MJ	0	WV	Alarm 8
①⑦	5	TP	XS7	CT12	" @SEGME" → 1st seg. no. word
	6	RA	PP1	FC24	Adv. add. label blk. → Add. seg. no. (3rd line)
	7	TU	A	PP11	Preset add. seg. no. (3rd line lab. blk.)
	10	RP	30006	PP12	} 3rd - 8th line lab. blk. → temps. Build Op. File IV this seg. and # sentences → "v" of index C5
	11	TP	[30000]	CT16	
	12	RJ	BF	BF1	
	13	RP	ZZ30000	PQ	} Program load III → core
	14	TP	DQ	PQ	
		CA	PP15		

Preface Section

	IA	PQ			
	0	TP	CT16	A	Octal segment no. → Av
	1	TJ	FC100	PQ7 <i>yes</i>	12 ₈ (10 ₁₀ ) → seg. no.?
	2	DV	FC100	Q	Divide seg. no. by 12 ₈ (NB-max. seg. no. = 63 ₁₀ )
	3	LQ	Q	6	Tens digit seg. no. left 6
	4	SA	Q	0	Two digit seg. no. → Av
	5	SA	FC101	6	Convert two digit ₁₀ seg. no. → XS3 and position in A
	6	MJ	0	PQ10	
	7	SA	FC11	14	Convert one digit ₁₀ seg. no. → XS3 and position in A
	10	AT	XS10	CT13	NTΔ [seg. no.] 0 → 2nd seg. no. word
⑱	11	RJ	OD	OD1	Sheet hdgs. (seg. no. & pg. no.) → output buffer
	12	SP	CT17	25	# blks Pref. (Term) → A in code-word position
	13	ZJ	PQ14 <i>yes</i>	PQ26 <i>no</i>	Is there Preface?
	14	AT	TC1	TH3	Codeword → G.T.H.
	15	RJ	TH2	TH	Read Preface from corr. prob. tape → input buffer
	16	RJ	HC	HC23	Preface hdgs → output buffer
⑲	17	TU	CT22	EC1	Init. running add. Preface → input edit routine
	20	TV	RC34	EF	Preset ent. add. for Pref. hdgs. (W/continued) in edit rtn.
	21	TU	RC	ED14	Preset input buff. add. → init. add. - 1 in edit rtn.
	22	RS	ED14	FC2	
	23	SP	CT23	0	# lines Preface → Av
	24	ST	FC1	CT2	# lines Preface - 1 → index C2
	25	RJ	EC	EC2	Edit Pref. and write on listing tape
⑲A	26	TP	CT21	Q	# lines partial blk. this segment → Q
	27	QT	FC23	A	# lines partial blk. segment + Preface → Qu
	30	TU	RC24	FR2	Preset initial add. Op. File IV (drum) in Op. File IV control routine
	31	TU	RC40	PR14	Preset add. File list → limiting add. initially
	32	ZJ	PR <i>yes</i>	PR1 <i>no</i>	Is there partial blk?
		CA	PQ33		

Sentence Section

	IA	PR			
	0	SP	FC2	6	Set blk. count = 1 in A in code- word position to count part. blk. # blks. (incl. part. blk.) seg. + Pref. → input fill buffer rtn. Fill input buffer
	1	AT	CT20	IR1	
	2	RJ	IR	IR2	
	3	TU	RC	ED14	}
	4	RS	ED14	FC2	
②①	5	IJ	CT5	PR10	yes Preset input buff. add. → init. add - 1 in edit rtn. Are there more sentences this segment?
②⑤	6	SP	CT17	25	yes
	7	ZJ	PR37	PS35	
	10	RJ	OD	OD2	Is there Termination? New page hdg. if required → out- put buffer
	11	RA	PR14	FC2	Adv. add. File list by 1 → add. next sent. no.
	12	TJ	RC40	PR14	Limit add. file list → current address?
	13	RJ	FR	FR1	Fill file list (core) from Op. file IV (drum)
②①	14	TP	[30000]	CT	XS3 sent. no. from file list → temp.
	15	RA	PR14	FC2	Adv. add. file list → add. of word with # lines & running add. of sent.
	16	TU	A	PR17	}
	17	TP	[30000]	A	
	20	ST	FC2	Q	# lines this sent. → Au; running add. this sent. → Av
	21	LQ	Q	17	}
	22	TU	Q	EC1	
	23	LQ	Q	6	Decrease # lines sent. by 1 → Au
	24	TV	Q	CT2	Running add. this sent. (or lib. rtn.) → input edit rtn.
②②	25	SJ	PR32	PR26	# lines this sent. (or lib. rtn.) → index C2 (+) ⇒ sentence ; (-) ⇒ library routine (Ck. left most bit of INFO word)
	26	TP	CT	XT3	Sent. no. → hdg.
	27	TV	RC4	EF	Preset add. sent. hdg. W/cont. in edit routine
	30	RJ	HC	HC33	Sent. hdgs. → output buffer
	31	MJ	0	PR35	
②③	32	TP	CT	XT47	Library routine name → hdg.
	33	TV	RC22	EF	Preset add. lib. rtn. hdg. W/cont. in edit. rtn.
	34	RJ	HC	HC51	Lib. rtn. hdgs. → output buffer

②4

35	RJ	EC	EC4	Edit sent. (or lib. rtn.) & write on listing tape
36	MJ	0	PR5	
37	AT	TC1	TH3	Codeword to tape handler
	CA	PR40		
	IA	PR40		
40	TU	5	PR41	Preset jn of repeat to trans. Dim. List → core
41	RP	[30000]	PS }	
42	TP	DD	DL }	Dimension list from drum → core
	CA	PR43		

Subscripted Variables (core) and Termination Section

	IA	PS			
	0	RJ	TH2	TH	Read Termination → input buffer
	1	RJ	OD	OD2	New page hdg. if required → out- put buffer
②6	2	SP	CT23	43	# lines Term. /2 = # subs. var. this seg. → A _L
	3	LT	0	A	# subs. var. this seg. → A _r
	4	ST	FC1	CT2	# subs. var. this seg.-1 → index
	5	TU	RC13	DS15	Preset add. initial array of this seg. in Term.
	6	TV	PS5	EV7	Preset Dim. List rtn. ref. → subs. var. (core) entry
	7	TU	6	DS21	Preset jn to search Dim. List. in Dim. List rtn.
	10	TV	RC11	EV32	Preset hdg. rtn. ref. → subs. var. (core) W/cont. entry
	11	RJ	HV	HV3	Init. subs. var. (core) hdgs. → Sect. hdg. blk.
	12	RJ	EV	EV1	Edit subs. var. (core) and write on listing tape
②7	13	TU	RC	ED14	} Preset input buff. add. → init. add.- 1 in edit rtn.
	14	RS	ED14	FC2	
	15	TP	XT14	XT11	Set section no. = zero in stored Term. hdgs.
	16	TV	RC10	EF	Preset add. for Term hdgs (W/cont) in edit rtn.
	17	TP	CT17	CT5	# blks. Term. → index C ₅
	20	TU	RC34	EC1	Init. running add. Term. buffer → input edit rtn.
②9	21	MJ	0	PS34	Adv. section no. by 1 New page hdg. if required → out- put buffer
	22	RA	XT11	FC1	
	23	RJ	OD	OD2	
	24	SP	CT23	0	# lines Termination → Av
	25	TJ	FC31	PS31	yes 171 _g > # lines Termination? (i.e. part. blk. Term. left?)
	26	ST	FC56	CT23	Decrease # lines Term. by 170 _g
	27	TP	FC42	CT2	# lines full blk. Term. - 1 → in- dex C ₂
	30	MJ	0	PS32	# lines partial blk. Term.- 1 → index C ₂
	31	ST	FC1	CT2	
	32	RJ	HC	HC41	Termination hdgs. → output buffer
	33	RJ	EC	EC2	Edit block of Termination and write on listing tape
②8	34	IJ	CT5	PS22	no All blks. Termination processed?
③0	35	RJ	IR	IR14	Locate next segment label blk.

36	RP	YY30000	PP	}
37	TP	DP	PP	}
	CA	PS40		

Program load II → core

End Listing Phase

	IA	PT			
③①	0	TV	CT6	PT1	Preset avail. add. output buffer
	1	TP	XS	[30000]	Fast feed 1 symbol → output buffer
	2	TV	CT6	PT5	Preset avail. add. output buff.
	3	RA	PT5	FC4	10g in "v" adv. → output buff. add. for "end of listing" blkt.
	4	RP	30004	PT6	
	5	TP	XT17	[30000]	END Δ OF ΔLISTING blkt. → output buffer
	6	RJ	BA	BA1	Adv. Output Buff. add. by 24 ₈ (20 ₁₀ ) in "u" and "v"
	7	RJ	BB	BB1	Terminate listing tape and rewind
	10	TP	TC7	TH3	
	11	RJ	TH2	TH	Rewind binary program tape
③②	12	TP	TC2	TH3	
	13	RJ	TH2	TH	Rewind corrected problem tape
	14	TP	FP10	UP3	Parameter → uniprint
	15	RJ	UP2	UP	Print: PROGRAM LISTING ON TAPE [-].
	16	TP	FP21	UP3	Parameter → uniprint
	17	RJ	UP2	UP	Print: COMPILATION COMPLETED
	20	MS	0	PT20	
		CA	PT21		

Build Op. File IV for Segment

		IA	BF		
	0	MJ	0	[30000]	
33	1	TP	TC4	TH3	Codeword → G.T.H.
	2	RJ	TH2	TH	Read 1 blk. Op. File III → file buffer
	3	TP	FB	A	1st word file buffer → A
	4	EJ	TL2	BF6	1st word file buffer = FILE△3? (Op. File III entry label)
	5	MJ	0	YP	Alarm 10
34	6	RP	34100	BF10	
	7	TP	ND	NL	Sentence No. (XS3) List → core
	10	TV	RC27	BF73	Preset init. add. Op. File IV (drum)
	11	TV	RC26	BF61	Preset init. add. statement buff. (core)
	12	TV	RC30	BG2	Preset init. add. routine buff. (core)
	13	TP	RC35	BG14	Preset init. add. routine file (drum)
35	14	TP	TC4	TH3	Codeword → G.T.H.
	15	RJ	TH2	TH	Read 1 blk. Op. File III → File buffer
	16	TP	FB	A	1st word File buffer = END△OF ?
	17	EJ	TL3	BG25	Yes ⇒ end Op. File III this segment
36	20	TU	BF16	BF21	Preset init. add. File Buff.
37	21	TP	30000	Q	Callword (or Z's) from File buff. → Q
	22	RA	BF21	FC2	1 in "u" adv. → ADD. of INFO. word assoc. W/callword
	23	SP	Q	0	Callword (or Z's) → Ar
	24	TJ	FC60	BG20	23000 > CW? (pseudo Op. sentence?)
	25	TJ	FC61	BF35	25000 > CW? (equat. for subs.var.?) NB → end of tape callword not in Op. File III
	26	TJ	FC62	BF37	26000 > CW? (equat. for non-subs. var.?)
38	27	TJ	FC63	BF52	30000 > CW? (statement of main prog.?)
	30	TJ	FC64	BF41	50000 > CW? (pseudo operation Hdg.?)
	31	TJ	FC65	BF45	60000 > CW? (library routine?)
	32	QJ	BG21-	BF76 +	(+) ⇒ 77---CW; (-) ⇒ word of Z's (end of information)
	33	TP	RC15	A	
	34	MJ	0	BG	
39	35	TP	RC16	A	
	36	MJ	0	BG	
40	37	TP	RC17	A	
		CA	BF40		

Build Op. File IV (cont.)

	IA	BF40		
	40	MJ	0	BG
④1	41	QT	FC50	A
				Designating bits of pseudo Op. CW → A
	42	LT	36	A
	43	SA	RC20	0
				Designating bits → "u" of Ar Add. base add. pseudo Op. sect. in sent. no. list
	44	MJ	0	BG1
④2	45	TU	BF21	BF46
	46	RA	[30000]	FC34
				Add. info. word → "u" of NI Lib. rtn. ind. (76 ₈ ) → Op. code of info. word
	47	LQ	Q	41
				Designating bits lib. rtn. C.W. → Qu
	50	TP	RC21	A
	51	MJ	0	BG
④3	52	TU	BF21	BF53
	53	TP	[30000]	A
	54	TJ	FC25	BF56 ^{no}
	55	MJ	0	BF76
	56	QT	FC54	A
	57	SA	RC15	0
				Add. info. word → "u" of NI Info. word → A Does info. word have "IP" flag? Yes ⇒ omit from file
	60	TU	A	BF61
				Last 3 digits of C.W. → Au Add. base address statements in sent. no. list
	61	TP	[30000]	[30000]
	62	RA	BF61	FC1
				Add. of XS3 sent. no. correspon- ding to CW → "u" of NI XS3 sent. no. → statement buffer
	63	TV	BF61	BF65
	64	TU	BF21	BF65
	65	TP	[30000]	[30000]
	66	AT	FC1	BF61
				Adv. add. in stmt. buff. by 1 in "v" Preset next add. stmt. buff. Info. word → stmt. buff. Information word → stmt. buff.
	67	TP	A	Q
	70	QT	FC32	A
	71	TJ	RC36	BF76 ^{no}
	72	RP	30170	BF74
	73	TP	SB	[30000]
	74	RA	BF73	FC56
				Adv. add. in stmt. buff. by 1 in "v" Next add. in stmt. buff. → Av Statement buffer full?
	75	TV	RC26	BF61
				Stmt. buff. → Op. File IV (drum) Adv. add. Op. File IV (drum) by 170 ₈ Preset add. stmt. buff. → init. add.
④5	76	RA	BF21	FC2
	77	TJ	RC23	BF21 ^{yes}
				Adv. Address file buff. by 1 in "u" More entries in file buff. to be processed?
100	MJ	0		BF14
	CA	BF101		

Build Op. File IV (cont.)

	IA	BG			
④⑥	0	QA	FC54	A	Base add. + last 3 digits CW = add. XS3 sent. no. → Au
④⑦	1	TU	A	BG2	
	2	TP	[30000]	[30000]	XS3 sent. no. → routine buffer
④⑧	3	RA	BG2	FC1	Adv. add. in routine buff. by 1 in "v"
	4	TU	BF21	BG6	Preset add. info. word
	5	TV	BG2	BG6	Preset add. routine buff.
	6	TP	[30000]	[30000]	Info. word → routine buff.
	7	AT	FC1	BG2	Adv. add. routine buff. by 1 in "v"
	10	TP	A	Q	
	11	QT	FC32	A	Next add. in routine buff. → Av
	12	TJ	RC37	BF76 no	Routine buff. full?
	13	RP	30170	BG15	
	14	[TP	RB	[30000]]	Routine buff. → routine file (drum)
④⑨	15	RA	BG14	FC56	Adv. add. routine file (drum) by 170g
	16	TV	RC30	BG2	Reset init. add. routine buffer
	17	MJ	0	BF76	
⑤⑩	20	ZJ	BF33	BF76	
⑤⑪	21	RJ	TH2	TH	Read 1 blk. Op. File III → file buffer
	22	TP	FB	A	} 1st word file buffer = END Δ OF ? Yes ⇒ end Op. file III this seg- ment
	23	EJ	TL3	BG25	
	24	MJ	0	YQ	Alarm 10
	25	RS	BF61	RC26	
	26	SA	FC57	17	
	27	TU	A	BG31	
	30	TV	BF73	BG32	
	31	RP	[30000]	BG33	
	32	TP	SB	[30000]	Part. stmt. buff. → Op. File IV (drum)
	33	RS	BG2	RC30	
	34	SA	FC57	17	
	35	TU	A	BG37	
	36	TV	BG14	BG40	
	37	RP	[30000]	BG41	
		CA	BG40		

Build Op. File IV (cont.)

	IA	BG40		
40	TP	RB	[30000]	Part. routine buff. → routine file (drum)
41	SP	BG32	0	Adv. Op. File IV → Av
42	SA	BF61	0	Adv. add. Op. file IV by # lines part. stmt. buff. → Av
43	TV	A	BG54	Preset add. Op. file IV
44	SS	FC1	17	Add. of info. word for last stmt. of seg. → Au
45	TU	A	BG46	Preset drum address of last stmt. info. word
46	RA	[30000]	FC24	Adv. # lines last stmt. Rtn. by 2 in "u" to count "Ip" and blank # lines routine file → Av
47	RS	BG14	RC35	# lines routine file + # lines part. buff. = total # lines routine file
50	AT	BG2	Q	jn to trans. routine file to Op. file IV → Au
51	SA	FC57	17	
52	TU	A	BG53	
53	RP	[30000]	BG55	
54	TP	RF	[30000]	Routine file (drum) → Op. file IV (drum)
55	RS	BG54	RC27	# lines Op. file IV (drum) before addition of routine file → Av
56	QA	FC32	A	# lines Op. file IV + # lines routine file = total # lines of Op. file IV → Q
57	LT	43	CT5	(# lines Op. file IV)/2 = # sentences this segment → "v" of C ₅
60	MJ	0	BF	
	CA	BG61		

Locate 1st Segment Label Blk. on Obj. Prog. Tape

	IA	LS		
⑤	0	MJ 0	[30000]	
	1	TP FC36	CT2	36g → index C ₂
	2	TU RC42	CT	Preset initial add. seg. lab. blk → 1st word list buffer
⑥	3	TP TC15	TH3	
	4	RJ TH2	TH	Read 1 blk. Object Prog. tape → list buffer
	5	TP LB	A	1st word list buffer → A
	6	EJ TL1	LS yes	(A) = SEGMENT ? (i.e. is blk. 1st seg. label blk.?)
	7	IJ CT2	LS4 no	37g blocks checked?
	10	MJ 0	BR10	Alarm 8
		CA LS11		

Check Label Corrected Prob. Tape

	IA	CL		
⑤③	0	MJ	0	[30000]
	1	TP	FC24	Q
	2	TU	RC42	CL3
⑤④	3	TP	[30000]	A
	4	RP	20006	CL6
	5	EJ	TL5	CL
	6	RA	CL3	FC2
	7	QJ	CL10	yes CL3 no
	10	MJ	0	YV
		CA	CL11	

Switch → Q (S.t. go back to begin loop 23g times)

Preset "u" of NI → Init. add. input buff.

Next word from corr. prob. title blkt. → A

Is this partial corr. prob. title? Adv. add. in title blkt. by 1 in "u"

Was this last word in title blkt.? Alarm 10

Page Number Routine

(57)

	IA	NP		
0	MJ	0	[30000]	
1	TP	CT11	Q	2nd page no. word → Q
2	QT	FC12	A	Last digit page no. → A
3	TJ	XS2	NP20 yes	9 in XS3 > last digit page no.?
4	RJ	NP4	[30000]	One shot jump (1st time → Pn15; succeeding times → NI)
5	QT	FC13	A	last two digits page no. → A
6	TJ	XS4	NP25 yes	99 in XS3 > last two digits page no.?
7	RJ	NP7	[30000]	One shot jump (1st time → Pn22; succeeding times → NI)
10	QT	FC14	A	Last two digits page no. → A
11	TJ	XS6	NP27 yes	999 in XS3 > last three digits page no.?
12	TP	XS17	CT10	△△△CON → 1st page no. word
13	TP	XS20	CT11	TINUED → 2nd page no. word
14	MJ	0	NP30	
15	TP	XS13	CT10	△△△△△P → 1st page no. word
16	TP	XS14	CT11	AGE△10 → 2nd page no. word
17	MJ	0	NP30	
20	RA	CT11	FC1	Adv. page no. by 1 ₈
21	MJ	0	NP30	
22	TP	XS15	CT10	△△△△△P A → 1st page no. word
23	TP	XS16	CT11	GE△1 0 0 → 2nd page no. word.
24	MJ	0	NP30	
25	RA	CT11	FC15	Advance next to last digit of page number by one octal and change last digit to zero in XS3
26	MJ	0	NP30	Advance third digit from right by one octal and change last two dig- its to zero in XS3
27	RA	CT11	FC16	Reset line count to four for page number line
30	TP	FC5	CT7	
31	MJ	0	NP	
	CA	NP32		

Op. File IV Control Routine

58

0	IA	FR		
1	MJ	0	[30000]	
2	RP	30170	FR3	} Fill file list in core from Op. File IV on drum
2	TP	[30000]	FL	
3	RA	FR2	FC55	Adv. Op. File IV drum add. by 170g in "u"
4	TU	RC25	PR14	Preset XS3 sent. no. add. → init. add. file list
5	MJ	0	FR	
	CA	FR6		

Dimension List Search Routine

	IA	DS		
	0	MJ	0	[30000]
	1	0	30000	0
	2	0	30000	30000
	3	0	30000	0
⑤9	4	TP	[30000]	DS1
	5	RA	DS4	FC2
	6	TU	A	DS7
	7	TP	[30000]	DS2
	10	AT	FC2	DS4
	11	TU	A	DS12
	12	SP	[30000]	0
	13	ST	DS1	DS3
	14	MJ	0	DS
⑥0	15	TP	[30000]	Q
	16	QT	FC23	DS1
	17	LQ	Q	17
	20	QT	FC23	CT15
	21	RP	[30000]	EP
	22	EJ	DL	DS23
⑥1	23	SN	Q	17
	24	SA	DS21	0
	25	SA	DS22	0
	26	TU	A	DS27
	27	TP	[30000]	DS2
	30	SA	FC2	0
	31	TU	A	DS32
	32	SP	[30000]	0
	33	ST	CT15	DS3
	34	TJ	FC77	DS44
	35	TJ	FC102	DS41
	36	RA	DS15	FC103
	37	RS	CT2	FC10
		CA	DS40	

} Output

Subs. var. (core/drum)  
 Add. in "u"  
 Subs. var. XS3 symbol  
 Modulus in "u"  
 Drum address → output  
 Adv. add. in Dim. List by 1 → add. XS3 symbol

XS3 symbol → output  
 Adv. add. in Dim List by 1 → add. for next drum add.

Drum add. for next array → Au  
 Drum add. next array-drum add.  
 curr. array=modulus → output  
 → Exit

Core add. of array → Qu, drum add. of array → Qv  
 Core add. of array → output

Drum address of array to "u" of A  
 Alarm 1  
 Is drum address in modified Dimension List?  
 - jn+r → "u" of A  
 +r → "u" of A  
 DL+r (add. of XS3 symbol) → Au  
 Preset NI  
 XS3 symbol → output  
 Adv. add. in dim. list → add. for next drum add.  
 Preset NI  
 Next drum add. in dim. list → Au  
 Next drum add.-drum add. current array = modulus → output  
 10000g > Modulus? Yes ⇒ 1 repeat "TP" in Termination  
 17777g > Modulus? Yes ⇒ 2 repeat "TP"'s in Termination  
 6 in "u" advance → address of next array transferred by Termination  
 Decrease index by 2 in "v"

	IA	DS40	
40	MJ	0	DS
41	RA	DS15	FC37
42	RS	CT2	FC1
43	MJ	0	DS
44	RA	DS15	FC24
45	MJ	0	DS
	CA	DS46	

61B

61C

Advance by 4 → address of next  
array trans. by Termination  
Decrease index by 1 in "v"

Advance by 2 → address of next  
array trans. by Termination

Input Buffer Routine

	IA	IR			
	0	MJ	0	[30000]	
	1	[0 [0	000]00	30000]	
⑥2	2	SP	IR1	0	
	3	TJ	FC71	IR10	yes
	4	ST	FC66	IR1	
	5	TP	TC12	TH3	
	6	RJ	TH2	TH	
	7	MJ	0	IR	
⑥3	10	AT	TC11	TH3	
	11	RJ	TH2	TH	
	12	TP	FC34	IR1	
	13	MJ	0	IR	
⑥4	14	TP	IR1	A	
	15	SJ	IR26	IR16	yes no
	16	SP	CT17	25	
	17	SA	IR1	0	
	20	AT	TC6	TH3	
	21	RJ	TH2	TH	
	22	TP	TC10	TH3	
⑥5	23	RJ	TH2	TH	
	24	TP	RC	CT	
	25	MJ	0	IR	
	26	SP	CT17	25	
	27	MJ	0	IR20	
		CA	IR30		

Input: # blks. seg. + pref. in  
codeword position  
# blks. seg. + pref. → A  
Max. # blks. input buff. + 1 > #  
blks. seg.+pref. still on tape?  
Decrease Blk. count in input line  
by # blks. full buffer  
Codeword to read Obj. Prog. tape  
→ G.T.H.  
Fill input buffer with blks. seg.  
+ pref.  
→ exit  
Codeword to read Obj. Prog. tape  
→ G.T.H.  
Remaining blks. of seg. +pref.  
→ input buff.  
Neg. no. → input line  
→ exit  
Input line → A  
All blks. seg. + pref. read to buff?  
(i.e. tape positioned for next seg.)  
# blks. term. → A in position for  
codeword  
# blks. term. + # blks. pref. not  
read to buff. → A  
Codeword to move forward obj. prog.  
tape → G.T.H.  
Move forward obj. prog. tape  
→ begin next seg. lab. blk.  
Codeword to read obj. prog. tape  
→ G.T.H.  
Read next segment label blk. → 1st  
blk. input buff.  
Preset add. 1st word label blk.  
→ Init. add. 1st blk. input buff.  
# blks. term. → A in position  
for codeword

Advance Output Buffer Address Routine

	IA	BA			
	0	MJ	0	[30000]	Exit
⑥⑥	1	RA	CT7	FC1	Adv. line count by 1 in "v" → next avail. line no.
	2	RA	CT6	FC6	Adv. output buff. add. by 24 ₈ (20 ₁₀ ) in "u" & "v"
	3	TJ	RC41	BA yes	Limiting output buff. add. > Current buff. add?
⑥⑦	4	RA	CT14	FC5	Adv. listing tape block count by # blks. (4) output buff.
	5	TJ	FC73	BA10 yes	2530 ₈ (1368 ₁₀ ) > curr. # blks. on listing tape?
	6	TV	RC7	OD4	Set switch ① → ② (end current listing tape at end next page)
	7	TP	FC	CT14	Listing Tp. blk. count = zero to render test on blk. count in- effective
⑥⑧	10	TP	TC16	TH3	Parameter → G.T.H.
	11	RJ	TH2	TH	Output Buffer → listing tape
	12	TP	RC1	CT6	Preset output buff. add. → init- ial value
	13	RP	10740	BA }	Fill output buff. W/XS3 space characters and exit
	14	TP	XS11	OB }	
		CA	BA15		

### Terminate Listing Tape Routine

	IA	BB			
⑥9	0	MJ	0	[30000]	Exit
	1	TV	CT6	BB2	Preset output buffer address
	2	TP	XT37	[30000]	Fast feed 1 & printer stop → output buffer
	3	RA	CT6	FC6	Adv. output buff. add. by 248 (20 ₁₀ ) in "u" & "v"
	4	ST	RC1	Q	# words in partial output buff. → "u" & "v" of Q
	5	QT	FC32	CT6	# words in partial output buff. → "v" of A & temp. 6
	6	TP	TC20	Q	Codeword to write 1 blk. output buff. → Q
	7	TJ	FC31	BB13	17lg > # words partial output buffer?
	10	RA	Q	FC72	Adv. count blks. in part. output buffer?
	11	RS	CT6	FC56	Decrease # words part. output buffer by 1
	12	MJ	0	BB7	
	13	TP	Q	TH3	Parameter → G.T.H.
	14	RJ	TH2	TH	Partial output buffer → listing tape
	15	TP	TC17	TH3	
	16	RJ	TH2	TH	Rewind listing tape
	17	MJ	0	BB	
		CA	BB20		

End Current Listing Tape at End of Page

	IA		BD		
⑦⑩ ①②	0	TV	CT6	BD1	Preset avail. output buff. add.
	1	TP	XS	[30000]	Fast feed 1 → output buffer
	2	TV	CT6	BD5	Preset avail. output buff. add.
	3	RA	BD5	FC5	Adv. output buff. add. → add. for MOUNT Δ NEXT Δ LISTING Δ TAPE, etc.
	4	RP	30014	BD6	} MOUNT Δ NEXT Δ LISTING Δ TAPE Δ ON Δ PRINTER., etc., → output buffer
	5	TP	XT23	[30000]	
	6	RJ	BA	BA1	Adv. output buff. add. by 24 ₈ (20 ₁₀ ) in "u" & "v"
	7	RJ	BB	BB1	Terminate current listing tape and rewind
	10	TP	FC	CT14	Reset count of blks. on listing tape = zero
	11	RJ	BA	BA12	Fill output buffer with XS3 space characters
⑦⑩	12	TP	FP31	UP3	} Type: CURRENT LISTING TAPE FULL. PUT NEW 1500 FT. TAPE ON SERVO ____. START TO CONTINUE LISTING.
	13	RJ	UP2	UP	
	14	MS	0	OD5	
		CA	BD15		

Output Control Subroutine

	IA	OC		
	0	MJ	0	[30000]
⑦①	1	IJ	CT2	OC10
	2	RJ	BA	BA1
	3	RA	OC	FC1
	4	TP	CT7	A
	5	TJ	FC67	OC21 no
	6	TV	RC6	OD2
	7	MJ	0	OC
⑦②	10	IJ	CT1	OC no
	11	RJ	BA	BA1
	12	TP	CT7	A
	13	TJ	FC70	OC17 no
	14	RA	OC	FC10
	15	RJ	OD	OD1
	16	MJ	0	OC
⑦③	17	RA	OC	FC11
	20	MJ	0	OC
⑦④	21	RJ	BA	BA1
	22	RJ	BA	BA1
	23	MJ	0	OC
		CA	OC24	

Are there quan. left this section?  
 No; adv. output buff. add. by  
 20₁₀ and line count by 1  
 Adv. exit add. by 1 in "v" ⇒ new  
 section  
 Line count → A  
 Was this 55th line on sheet or  
 beyond when new section next  
 Set switch (A) → (A2)

Was this last entry in blkt?  
 No ⇒ same blkt. - same sheet exit  
 Yes; adv. Output buff. add. by  
 20₁₀ and line count by 1  
 Line count → A  
 Was this 63rd line on sheet or  
 beyond when same section next  
 Yes; adv. exit add. by 2 in "v"  
 ⇒ new sheet exit  
 New page heading → output buffer  
 → Exit

Adv. exit add. by 3 in "v" ⇒ new  
 blkt.-same sheet exit

Adv. output buffer by 20₁₀ and  
 line count by 1 (space blkt.)  
 Adv. output buffer by 20₁₀ and  
 line count by 1 (space blkt.)

Page Heading Control Subroutine

	IA	OD		
	0	MJ	0	[30000]
(75)	1	TV	RC6	OD2
(75A)	2	RJ	OD2	[OD3]
(76) (A1)	3	MJ	0	OD
(77) (A2)	4	RJ	OD4	[OD5]
(78) (B1)	5	RJ	NP	NP1
	6	TV	CT6	OD10
	7	RP	30002	OD11
	10	TP	CT12	[30000] }
	11	RA	OD10	FC17
	12	TV	A	OD14
	13	RP	30002	OD15
	14	TP	CT10	[30000] }
(79)	15	RJ	BA	BA1
	16	RJ	BA	BA1
	17	RJ	BA	BA1
	20	MJ	0	OD
		CA	OD21	

Set switch (A) → (A2)  
Switch (A)

Switch (B) B1 = OD5  
B2 = BD

Adv. page no.  
Preset add. 1st seg. no. word =  
avail. output buff. add.

Segment no. words → sheet hdg.  
blkt.  
22g in "v" adv. → add. for 1st  
page no. word  
Preset add. for 1st page no. word

Page no. words → sheet hdg. blkt.  
Adv. output buff. add. by 20₁₀ and  
line count by 1 (Sheet hdg. blkt.)  
Adv. output buff. add. by 20₁₀ and  
line count by 1 (Space blkt.)  
Adv. output buff. add. by 20₁₀ and  
line count by 1 (Space blkt.)

Convert Octal Address to XS3

	IA	CA		
	0	MJ	0	[30000]
	1	0	30000	30000
	2	[0	30000	30000]
⑧0	3	RJ	CA30	CA22
	4	LT	6	CA1
	5	MJ	0	CA
⑧1	6	RJ	CA30	CA22
	7	LT	0	CA1
	10	MJ	0	CA
⑧2	11	SP	FC26	6
	12	RJ	CA30	CA23
	13	LT	0	CA1
	14	MJ	0	CA
⑧3	15	RJ	CA30	CA22
	16	SA	XS5	6
	17	LT	0	CA1
	20	TP	XS3	CA2
	21	MJ	0	CA
⑧4	22	TP	FC	A
⑧5	23	TP	FC34	CT3
	24	LQ	CA2	3
	25	QA	RC43	A
	26	SA	FC74	6
	27	LQ	CT3	1
	30	QJ	CA24 no	[30000]yes
		CA	CA31	

Output = XS3 address  
Input = Octal address in "u"  
W/zero (octal) fill  
Convert address  
XS3 add. W/octal zeros on right  
→ output

XS3 address W/octal zeros on left  
→ output

XS3 hyphen → rightmost digits A_L

Converted address W/hyphen left  
→ output

Converted address → A_L packed right

Add. close parent. following XS3 address  
XS3 address W/close parent. → 1st output  
Open parent. → 2nd output

Zeroize A  
Set index = 4  
Next digit octal input add.  
→ Qop  
Add. next digit to be converted → Aop  
Convert digit to XS3 and shift → A_L

Convert Address

All 5 digits converted? Yes;  
sub-exit. XS3 address in "A"  
left packed right

Convert Octal Word to XS3

	IA	CW		
	0	MJ	0	[30000]
	1	0	30000	30000
	2	0	30000	30000
	3	0	30000	30000
86	4	0	30000	30000
	5	TP	FC47	CW1
	6	TP	FC25	CW2
	7	TP	FC25	CW3
	10	TP	RC14	CW20
	11	TV	FC35	CW17
	12	TP	FC46	Q
87	13	RS	CW17	FC27
	14	SP	CW4	3
	15	LT	10000	CW4
	16	LT	0	A
	17	SA	FC11	[30000]
	20	[AT	CW1	CW1]
	21	QJ	CW22	yes CW13 no
	22	QJ	CW23	no CW yes
	23	RA	CW20	FC3
	24	TV	FC36	CW17
	25	MJ	0	CW13
		CA	CW26	

Output - XS3 Op. code  
 Output - XS3 "u" add.  
 Output - XS3 "v" add.  
 Input - octal computer word  
 $\Delta\Delta\Delta\Delta$  W/zero fill  $\rightarrow$  1st word output  
 $\Delta$  W/zero fill  $\rightarrow$  2nd word output  
 $\Delta$  W/zero fill  $\rightarrow$  3rd word output  
 Preset add. 1st output word  
 Preset shift count  $\rightarrow$  14g  
 switch  $\rightarrow$  Q 

30	30200	00000
----	-------	-------

  
 Decrease shift count by 6  
 Next octal digit input word  $\rightarrow$  1st digit  $A_L$   
 Shifted input word ( $A_r$ )  $\rightarrow$  input line  
 Digit to be converted  $\rightarrow$  rightmost digit  $A_r$   
 Conv. octal digit  $\rightarrow$  XS3 and shift to position in A  
 Converted digit  $\rightarrow$  output word  
 Output word full?  
 Entire octal input word converted?  
 1 in "u" & "v" adv.  $\rightarrow$  add. next output word  
 Reset shift count  $\rightarrow$  36g

Heading Rtn. for Const. Pool, Preface, Sentence and Termination

	IA	HC		
88	0	MJ	0	[30000]
	1	TP	XS70	XS65
	2	TV	CT6	HC4
89	3	RP	30003	HC11
	4	TP	XS63	[30000]
	5	TP	XS71	XS65
	6	TV	CT6	HC10
	7	RP	30005	HC11
90	10	TP	XS63	[30000]
	11	RJ	OC	OC21
	12	TV	CT6	HC14
	13	RP	30002	HC15
	14	TP	XS56	[30000]
	15	RJ	BA	BA1
	16	TV	CT6	HC20
	17	RP	30002	HC21
	20	TP	XS61	[30000]
	21	RJ	OC	OC21
91	22	MJ	0	HC
	23	TP	XS76	XS73
	24	TV	CT6	HC26
92	25	RP	30002	HC11
	26	TP	XS72	[30000]
	27	TP	XS77	XS73
	30	TV	CT6	HC32
93	31	RP	30004	HC11
	32	TP	XS72	[30000]
94	33	TV	CT6	HC35
	34	RP	30004	HC11
	35	TP	XT	[30000]
	36	TV	CT6	HC40
	37	RP CA	30006 HC40	HC11

Setup const. pool sect. hdg. w/o continued

Preset add. sect. hdg. blkt = output buffer

CONSTANT Δ POOL → sect. hdg. blkt. Setup Const. Pool sect. hdg. W/ continued

Preset add. sect. hdg. blkt. = output buffer

CONSTANT Δ POOL ← CONTINUED → sect. hdg. blkt.

Adv. output buff. add. by 40₁₀ (50₈)

ADDRESS → col. hdg. blkt.

Adv. output buff. add. by 20₁₀ (24₈)

Underscores → output buffer

Adv. output buff. add. by 40₁₀ (50₈)

Setup Preface sect. hdg. W/O continued

Preset add. sect. hdg. blkt = output buff. add.

PREFACE → sect. hdg. blkt.

Setup pref. sect. hdg. W/continued

Preset add. sect. hdg. blkt. = output buff. add.

PREFACE ← CONTINUED → sect. hdg. blkt.

Preset add. sect. hdg. blkt. = output buff. add.

SENTENCE NUMBER [----] → sect. hdg. blkt.

Preset add. sect. hdg. blkt. = output buff. add.

Heading Routine (Cont.)

	IA	HC40		
	40	TP	XT [30000]	SENTENCE NUMBER[----] CONTINUED → sect. hdg. blkt.
95	41	TV	CT6 HC44	Preset add. sect. hdg. blkt. = Output buffer add.
	42	TP	XT15 XT12	
	43	RP	30005 HC11	
	44	TP	XT6 [30000]	TERMINATION → sect. hdg. blkt.
96	45	TV	CT6 HC50	Preset add. sect. hdg. blkt = Output buffer add.
	46	TP	XT16 XT12	Setup stored hdg. W/continued
	47	RP	30006 HC11	TERMINATION (SECTION--) --
	50	TP	XT6 [30000]	CONTINUED → <i>section</i> hdg. blkt.
97	51	TP	XT52 XT50	Set up library routine hdg. W/O continued
	52	TV	CT6 HC54	Preset add. sect. hdg. blkt. = Output buffer add.
	53	RP	30005 HC11	
	54	TP	XT44 [30000]	LIBRARY ROUTINE ----- → sect. hdg. blkt.
98	55	TP	XT53 XT50	Set up lib. rtn. hdg W/continued
	56	TV	CT6 HC60	Set add. sect. hdg. blkt. = output buffer add.
	57	RP	30006 HC11	
	60	TP	<del>XT44</del> [30000]	LIBRARY ROUTINE [-----] -- CONTINUED → sect. hdg. blkt.
		CA	HC61	

### Heading Routine for Subscripted Variables

	IA	HV			
99	0	MJ	0	[30000]	
	1	TP	XS30	XS25	Set up section hdg. for drum initial
100	2	MJ	0	HV4	
	3	TP	XS32	XS25	Set up section hdg. for core initial
101	4	TV	CT6	HV6	Set add. sect. hdg. blkt. = Output buffer add.
	5	RP	30005	HV15	
	6	TP	XS21	[30000]	SUBSCRIPTED VARIABLES [ DRUM CORE ]
102	7	TP	XS31	XS25	→ section hdg. blkt. Set up section hdg. for drum W/continued
	10	MJ	0	HV12	
103	11	TP	XS33	XS25	Set up section hdg. for Core W/continued
104	12	TV	CT6	HV14	Set add. sect. hdg. blkt = Output buffer add.
	13	RP	30007	HV15	
	14	TP	XS21	[30000]	SUBSCRIPTED VARIABLES [ DRUM CORE ]
105	15	TU	RC4	HV23	— CONTINUED → out. buffer Preset "u" of TP → add. of stored col. hdg.
	16	RJ	OC	OC21	Two space blkts → Output Buffer
	17	TP	FC20	Q	Switch [ 06   00000   00000 ] → Q
	20	TV	CT6	HV23	Preset add. col. hdg. blkt. = Output buffer add.
	21	RA	HV23	FC1	1 in "v" adv. → add. 1st col. hdg.
	22	RP	30004	HV24	
	23	TP	[30000]	[30000]	Column hdg. (or underscores) → out. put buffer
	24	RA	HV23	FC21	5 in "v" adv. → add. for next col. hdg. (or underscores)
	25	QJ	HV26 ^{yes}	HV22 ^{no}	All col. hdgs. (or underscores) → output buffer?
	26	QJ	HV27 ^{no}	HV33 ^{yes}	Underscores transferred yet?
	27	RJ	BA	BA1	Adv. output buff. add. by 20 ₁₀ (24g)
	30	TU	RC5	HV23	Preset "u" of TP → add. of stored underscores
	31	TP	FC22	Q	Switch [ 04   00000   00000 ] → Q
106	32	MJ	0	HV20	
	33	RJ	OC	OC21	Two space blkts. → output buffer
	34	MJ	0	HV	→ Exit
		CA	HV35		

Edit Subs. Var. and Write on Listing Tape

	IA	EV		
107	0	MJ	0	[30000]
	1	TV	CT6	EV12
	2	RS	EV12	FC5
108	3	TP	FC11	CT1
	4	RA	EV12	FC21
	5	SA	FC1	0
	6	TV	A	EV13
	7	RJ	DS	[30000]
109	10	SP	XS11	44
	11	SA	DS2	22
	12	LT	0	[30000]
	13	TP	A	[30000]
	14	RA	EV13	FC1
	15	TV	A	EV20
110	16	TP	DS1	CA2
	17	RJ	CA	CA6
	20	TP	CA1	[30000]
	21	RS	DS1	FC2
	22	AT	DS3	CA2
111	23	RJ	CA	CA11
	24	RA	EV20	FC1
	25	TV	A	EV26
	26	TP	CA1	[30000]
	27	RJ	OC	OC1
	30	MJ	0	EV4
	31	MJ	0	EV
	32	RJ	HV	[30000]
	33	MJ	0	EV1
		CA	EV34	

Set assem. blkt. add. = Output buff. add.  
 Dec. assem. blkt. add. → Preset for 1st sym.  
 Preset index  $C_1$  → # variables/ blkt. - 1  
 Adv. assem. blkt. add. → add. 1st part next sym.  
 1 in "v" adv. → add. last part next symbol  
 Preset add. for last part symbol  
 Dimension list rtn. { DS1 = Add. of array in "u"  
 DS2 = XS3 sym.  
 DS3 = modulus  
 -1 in "u"  
 XS3 space char. →  $A_L$   
 1st part XS3 sym. →  $A_1$ ; last part sym. →  $A_r$   
 1st part XS3 sym. → output buff. (assem. blkt.)  
 Last part XS3 sym. → output buff. (assem. blkt.)  
 1 in "v" adv. → add. for initial add. entry  
 Preset "u" of TP → add. for initial add. entry  
 Initial add. for array → conv. routine  
 Conv. add. (CA) W/O hyphen. (CA1 = XS3 add. packed to right)  
 Add. entry → assem. blkt.  
 Init. add. of array - 1 →  $A_u$   
 Final add. of array → conv. routine  
 Conv. address (CA) W/hyphen (CA1 = XS3 add. packed to right)  
 Adv. assem. blkt. add. by 1 in "v"  
 Next assem. blkt. add. → "v" of TP  
 Last address for array → assem. blkt.  
 → Output control  
 Same blkt.; same sheet  
 New section  
 New blkt.; new sht. → heading W/continued → output buff.  
 New blkt.; same sht.

Edit Coding or Constants Routine

	IA	EC			
	0	MJ	0	[30000]	Exit next section
	1	O	[30000]	0	Initial running add. this section
113	2	TP	RC32	ED11	Set MJ to by-pass TJ
	3	MJ	0	EC5	
114	4	TP	RC31	ED11	Set TJ for input buff. check
115	5	TV	CT6	EC16	Preset add. assem. blkt. = output buff. add.
	6	RA	EC16	FC1	Adv. assem. blkt. add. → add. for add. entry
	7	TP	EC1	Q	Running add. 1st word this section
	10	QT	FC33	CT15	Last digit initial add. → "u" temp. 2 and A
116	11	TP	EC1	CA2	Init. add. → input conv. routine
	12	ZJ	EC13	no EC37	yes Last digit init. add. = zero?
	13	RJ	CA	CA15	Convert octal address W/parents.
	14	TV	CT6	EC15	
	15	TP	CA2	[30000]	Open parent. (if 5 digit add. W/parents) or zeros → output buffer
117	16	TP	CA1	[30000]	XS3 address → output buffer
	17	TP	CT15	A	Last octal digit initial add. → A
	20	TV	CT6	ED17	Preset assem. blkt. add. = output buffer
	21	TJ	FC37	EC25	yes 4 > last digit init. add.?
	22	TP	FC	CT4	Zero → blkt. index C ₄ (i.e. odd line ⇒ next line has add.)
	23	SS	FC37	0	Dec. last digit of add. by 4 in "u" → A
	24	MJ	0	EC26	
	25	TP	FC1	CT4	Set blkt. index C ₄ → 1 (i.e. even line ⇒ next line has no add.)
	26	TP	EC1	Q	Init. add. from input → Qu
	27	TJ	FC40	EC33	yes 3 > last digit add.?
	30	RA	ED17	FC35	Adv. assem. blkt. add. → Preset for last entry in blkt.
118	31	TP	FC	CT1	Preset index for 1 entry in blkt.
	32	MJ	0	ED6	
119	33	TJ	FC24	ED	yes 2 > last digit add.?
	34	RA	ED17	FC76	Adv. assem. blkt. add. → preset for 3rd entry in blkt.
	35	TP	FC1	CT1	Set index for 2 entries in blkt.
	36	MJ	0	ED6	
	37	RJ	CA	CA3	Convert octal address W/octal zeros on right
	40	MJ	0	EC16	
		CA	EC41		

Edit Coding or Const. (Cont.)

120	0	IA	ED						
	1	TJ	FC2	ED4					1 > last digit add.?
		RA	ED17	FC10					Adv. assem. blkt. add. → Preset for 2nd entry in blkt.
	2	TP	FC10	CT1					Set index for 3 entries in blkt.
121	3	MJ	0	ED6					
	4	RS	ED17	FC11					Dec. assem. blkt. add. → Preset for add. 1st entry in blkt.
122	5	TP	FC11	CT1					Set index for 4 entries in blkt.
123	6	QT	FC41	CT15					1st four digits init. add. → Temp. 2 (add. ctr.)
124	7	RA	ED17	FC21					Adv. assem. blkt. add. by 5 in "v" → add. for next entry
125	10	RA	ED14	FC2					Adv. input buff. add. by one
	11	[TJ	RC2	ED14]					Limiting value > input buff. add.
	12	RJ	IR	IR2					Fill input buff. from tape
	13	TU	RC	ED14					Preset input buff. add. → initial add.
	14	TP	[30000]	CW4					Next word from input buff. → Conv. routine.
126	15	RJ	CW	CW5					Convert octal word → XS3
	16	RP	30003	ED20					
	17	TP	CW1	[30000]					XS3 entry ( 3 words) → assem. blkt.
	20	RJ	OC	OC1					⇒ Output control
	21	MJ	0	ED7					⇒ same blkt.-same sheet
	22	MJ	0	EC					New section
	23	MJ	0	EF					New blockette-new sheet
127	24	IJ	CT4	ED46					New blockette-same sheet ⇒ add. entry required next blkt?
	25	RA	CT15	FC43					Yes; adv. ctr. (temp. 2) by 10 ₈ in "u"
	26	TP	FC75	Q					Mask → Q
	27	QT	A	A					Last 2 digits add. → A
	30	ZJ	ED40	ED31	no	yes			Last two digits add. = zero?
	31	RJ	BA	BA1					Adv. output buff. add. by 20 ₁₀ (24 ₈ ) (space blkt. → buff.)
128	32	RJ	BA	BA1					Space blkt. → output buffer
	33	TP	CT7	A					Line count → A
	34	TJ	FC70	ED40		no			Was 2nd space blkt. 63rd line on sheet or beyond
	35	RJ	OD	OD1					New pg. hdg. blkt. → output buff. and reset line count = 4.
	36	TV	EF	ED37					
	37	RJ	HC	[30000]					Sect. hdgs. W/continued → output buffer
		CA	ED40						

Edit Coding or Const. (Cont.)

	IA	ED40			
(129)	40	TV	CT6	ED44	Preset add. assem. blkt. = output buff. add.
	41	RA	ED44	FC1	Adv. assem. blkt. add. → add. for XS3 add. entry
	42	TP	CT15	CA2	Octal address → input conv. routine
	43	RJ	CA	CA3	Convert octal address W/octal zero on right
	44	TP	CA1	[30000]	XS3 address entry → output buffer
	45	TP	FC1	CT4	Set blkt. index $C_4$ → 1 (No add. entry next blkt.) ⁴
(130)	46	TV	CT6	ED17	Preset assem. blkt. add. = output buffer add.
	47	RS	ED17	FC11	Dec. assem. blkt. add. → preset for add. 1st entry in blkt.
	50	TP	FC11	CT1	Preset index $C_1$ → # entries/blkt.
	51	MJ	0	ED7	
		CA	ED52		

Edit Coding or Const. (new blkt.-New Sheet Sect.)

(131)	0	IA	EF		
		RJ	HC	[30000]	Section heading W/continued → Output buffer
	1	IJ	CT4	ED46	no Add. entry required next blkt?
	2	RA	CT15	FC43	Adv. address counter (temp. 2) by $10_8$ in "u"
	3	MJ	0	ED40	
		CA	EF4		

Edit XS3 Variable Symbol for 77's

(132)

	IA	ES		
0	MJ	0	[30000]	
1	0	30000	30000	Output-XS3 symbol packed right W/0 ₈ fill
2	0	30000	30000	Input-XS3 symbol packed left W/77 ₈ fill
3	TP	FC21	CT3	Set index C ₃ = 5
4	TP	FC	ES1	Zero → output line
5	LQ	ES2	6	Next XS3 symbol → rightmost digits of Q
6	QT	FC12	A	Next XS3 symbol → rightmost digits of A _L
7	TJ	FC12	ES11	yes
10	MJ	0	ES	77 ₈ > symbol?
11	LQ	ES1	6	Exit on first 77 ₈ encountered
12	AT	ES1	ES1	
13	IJ	CT3	ES5	no
14	MJ	0	ES	Symbol → rightmost digits output
	CA	ES15		All XS3 char. of symbol checked?

Tape Handler Codewords

	IA	TC				
0	5 0	001	05	LB	Read 1 blk. corrected prob. tape → list buffer	
1	5[0	000]	05	IB	Read [0000] blks. corrected prob. tape → input buffer	
2	1 0	000	05	0	Rewind corrected prob. tape	
3	3[0	000]	05	0	Move forward [0000] blks. corrected <b>prob.</b> tape	
4	5 0	001	05	FB	Read 1 blk. corrected prob. tape → file buffer	
5	5[0	000]	05	DL	Read [0000] blks. corrected prob. tape → dim. list region	
6	3[0	000]	[77]	0	Move forward [0000] blks. obj. prog. tape	
7	1 0	000	[77]	0	Rewind obj. prog. tape	
10	5 0	001	[77]	IB	Read 1 blk. obj. prog. tape → In- put buffer	
11	5[0	000]	[77]	IB	Read [0000] blks. obj. prog. tape → input buffer	
12	5 0	011	[77]	IB	Read 9 blks. obj. prog. tape → input buffer	
13	5 0	007	[77]	LB	Read 7 blks. obj. prog. tape → list buffer	
14	5 0	005	[77]	LB	Read 5 blks. obj. prog. tape → list buffer	
15	5 0	001	[77]	LB	Read 1 blk. obj. prog. tape → list buffer	
16	7 4	204	[77]	OB	Write 4 blks. output buff. on listing tape	
17	1 0	000	[77]	0	Rewind listing tape	
20	7 4	201	[77]	OB	Write 1 blk. output buff. on listing tape	
21	7 7	777	03	77777	Object prog. Uniservo 3 for 5 Servo layout	
22	7 7	777	06	77777	Object prog. Uniservo 6 for 7 Servo layout	
23	5[0	000]	05	LB	Read [0000] blks. corrected prob. tape → list buffer	
	CA	TC24				

Tape Labels (XS3)

	IA	TL								
0	74	74747	47474	Z	Z	Z	Z	Z	Z	Word of Z's
1	65	30324	73050	S	E	G	M	E	N	
2	31	34463	00106	F	I	L	E	△	3	
3	30	50270	15131	E	N	D	△	O	F	
4	65	73472	55146	S	Y	M	B	O	L	
5	67	50342	65127	U	N	I	C	O	D	} Source program labels
6	01	67503	42651	△	U	N	I	C	O	
7	01	01675	03426	△	△	U	N	I	C	
10	26	51273	00152	C	O	D	E	△	P	
11	34	26512	73001	I	C	O	D	E	△	
12	50	34265	12730	N	I	C	O	D	E	
13	26	51506	56624	C	O	N	S	T	A	
	CA	TL14								

XS3 Codes

	IA	XS								
0	37	01010	10101	@	△	△	△	△	△	Fast feed 1 symbol
1	00	00000	00004	08	08	08	08	08	1x	
2	00	00000	00014	08	08	08	08	08	9	
3	00	00000	00017	08	08	08	08	08	(x	XS3 open parent
4	00	00000	01414	08	08	08	08	9	9	
5	43	00000	00000	)x	08	08	08	08	08	XS3 close parent
6	00	00001	41414	08	08	08	9	9	9	
7	37	65303	24730	@	S	E	G	M	E	} Segment number setup
10	50	<del>66</del> 010	0	N	T	△	-	-	-	
11	01	01010	10101	△	△	△	△	△	△	} Page number setups
12	52	24323	00104	P	A	G	E	△	1	
13	01	01010	10152	△	△	△	△	△	P	} Subscripted Variables Section Heading
14	24	32300	10403	A	G	E	△	1	0	
15	01	01010	15224	△	△	△	△	P	A	} Subscripted Variable
16	32	30010	40303	G	E	△	1	0	0	
17	01	01012	65150	△	△	△	C	O	N	} Column Headings
20	66	34506	73027	T	I	N	U	E	D	
21	01	65672	56526	△	S	U	B	S	C	
22	54	34526	63027	R	I	P	T	E	D	
23	01	70245	43424	△	V	A	R	I	A	
24	25	46306	50117	B	L	E	S	△	(	
25	0	0	0	[	-	-	-	-	-	
26	02	26515	06634	-	C	O	N	T	I	
27	50	67302	70101	N	U	E	D	△	△	
30	27	54674	74301	D	R	U	M	)	△	
31	27	54674	74302	D	R	U	M	)	-	
32	26	51543	04301	C	O	R	E	)	△	
33	26	51543	04302	C	O	R	E	)	-	
34	01	01016	57347	△	△	△	S	Y	M	
35	25	51460	10101	B	O	L	△	△	△	
36	01	01242	72754	△	△	A	D	D	R	
37	30	65653	06501	E	S	S	E	S	△	
	CA	XS40								

XS3 Codes

	IA	XS40								
40	01	01010	20202	△	△	△	-	-	-	} Subscripted Variable Underscores
41	02	02020	10101	-	-	-	△	△	△	
42	01	02020	20202	△	△	-	-	-	-	
43	02	02020	20201	-	-	-	-	-	△	} Non-Subscripted Variables Section Heading Setups for Preceding Heading
44	01	50515	00265	△	N	O	N	-	S	
45	67	25652	65434	U	B	S	C	R	I	
46	52	66302	70170	P	T	E	D	△	V	
47	24	54342	42546	A	R	I	A	B	L	
50	0	0	0	[-	-	-	-	-	-]	
51	50	66345	06730	N	T	I	N	U	E	
52	27	01010	10101	D	△	△	△	△	△	
53	30	65010	10101	E	S	△	△	△	△	
54	30	65020	22651	E	S	-	-	C	O	
55	65	73472	55146	S	Y	M	B	O	L	} Non-Subscripted Variable column heading
56	01	01010	10124	△	△	△	△	△	A	
57	27	27543	06565	D	D	R	E	S	S	
60	02	02020	20202	-	-	-	-	-	-	} Non-Subscripted Variable Underscores
61	01	01010	10102	△	△	△	△	△	-	
62	02	02020	20202	-	-	-	-	-	-	
63	01	26515	06566	△	C	O	N	S	T	} Constant Pool section heading
64	24	50660	15251	A	N	T	△	P	O	
65	0	0	0	[-	-	-	-	-	-]	
66	50	66345	06730	N	T	I	N	U	E	
67	27	01010	10101	D	△	△	△	△	△	
70	51	46010	10101	O	L	△	△	△	△	} Setups for Const. Pool Sect. Hdg.
71	51	46020	22651	O	L	-	-	C	O	
72	01	52543	03124	△	P	R	E	F	A	} Preface Section Heading
73	0	0	0	[-	-	-	-	-	-]	
74	50	66345	06730	N	T	I	N	U	E	
75	27	01010	10101	D	△	△	△	△	△	} Setups for Preface Section Heading
76	26	30010	10101	C	E	△	△	△	△	
77	26	30020	22651	C	E	-	-	C	O	
	CA	XS100								

XS3 Codes (Cont.)

	IA	XT			
0	01	65305	06630	△ S E N T E	} Sentence Section Heading
1	50	26300	15067	N C E △ N U	
2	47	25305	40101	M B E R △ △	
3	0	0	0	[Sent. no. in std. form]	
4	02	02265	15066	- - C O N T	} Termination Section Headings
5	34	50673	02701	I N U E D △	
6	01	66305	44734	△ T E R M I	
7	50	24663	45150	N A T I O N	} Setups for Preceding heading
10	01	17653	02666	△ ( S E C T	
11	0	0	0	[Section No.]	} Printer Stop Symbol
12	0	0	0	[- - - - -]	
13	66	34506	73027	T I N U E D	} Printer Stop Symbol
14	34	51500	10003	I O N △ O ∅	
15	43	01010	10101	) △ △ △ △ △	} Printer Stop Symbol
16	43	02022	65150	) - - C O N	
17	01	01010	10130	△ △ △ △ △ E	} Printer Stop Symbol
20	50	27015	13101	N D △ O F △	
21	46	34656	63450	L I S T I N	} Printer Stop Symbol
22	32	01010	10101	G △ △ △ △ △	
23	01	01475	16750	△ △ M O U N	} Printer Stop Symbol
24	66	01503	07266	T △ N E X T	
25	01	46346	56634	△ L I S T I	} Printer Stop Symbol
26	50	32016	62452	N G △ T A P	
27	30	01515	00152	E △ O N △ P	} Printer Stop Symbol
30	54	34506	63054	R I N T E R	
31	22	01275	10150	. △ D O △ N	} Printer Stop Symbol
32	51	66012	63324	O T △ C H A	
33	50	32300	15251	N G E △ P O	} Printer Stop Symbol
34	65	34663	45150	S I T I O N	
35	01	51310	15224	△ O F △ P A	} Printer Stop Symbol
36	52	30542	20101	P E R . △ △	
37	37	60606	06060	Σ Σ Σ Σ Σ Σ	} Printer Stop Symbol
	CA	XT40			

	IA	XT40								
40	01	01010	10152	△	△	△	△	△	P	} Heading for Listing
41	54	51325	42447	R	O	G	R	A	M	
42	01	46346	56634	△	L	I	S	T	I	
43	50	32010	10101	N	G	△	△	△	△	} Library Routine Heading
44	01	46342	55424	△	L	I	B	R	A	
45	54	73015	45167	R	Y	△	R	O	U	
46	66	34503	00101	T	I	N	E	△	△	} Setups for Lib. Routine Heading
47	0	0	0	[	Routine name.	]				
50	0	0	0	[	-	-	-	-	-]	
51	66	34506	73027	T	I	N	U	E	D	} Setups for Lib. Routine Heading
52	01	01010	10101	△	△	△	△	△	△	
53	01	02022	65150	△	-	-	C	O	N	
	CA	XT54								

### Flexowriter Printout

IA	FP								
0	00	FP1	7						Codeword
1	01	01010	10101	△	△	△	△	△	
2	52	24656	50170	P	A	S	S	△	V
3	22	01010	10101	.	△	△	△	△	△
4	01	46346	56634	△	L	I	S	T	I
5	50	32015	13101	N	G	△	O	F	△
6	52	54513	25424	P	R	O	G	R	A
7	47	22010	10101	M	.	△	△	△	△
10	00	FP11	10						Codeword
11	01	01010	10101	△	△	△	△	△	△
12	01	01010	10101	△	△	△	△	△	△
13	01	01010	10101	△	△	△	△	△	△
14	01	52545	13254	△	P	R	O	G	R
15	24	47014	63465	A	M	△	L	I	S
16	66	34503	20151	T	I	N	G	△	O
17	50	01662	45230	N	△	T	A	P	E
20	0	0	0	[					]
21	00	FP22	7						Servo # Codeword
22	01	01010	10101	△	△	△	△	△	△
23	01	01010	10101	△	△	△	△	△	△
24	01	01010	10101	△	△	△	△	△	△
25	01	26514	75234	△	C	O	M	P	I
26	46	24663	45150	L	A	T	I	O	N
27	01	26514	75246	△	C	O	M	P	L
30	30	66302	72201	E	T	E	D	.	△
31	00	FP32	23						Codeword
32	01	01010	10101	△	△	△	△	△	△
33	01	01010	10101	△	△	△	△	△	△
34	01	01010	10101	△	△	△	△	△	△
35	01	26675	45430	△	C	U	R	R	E
36	50	66014	63465	N	T	△	L	I	S
37	66	34503	20166	T	I	N	G	△	T
	CA	FP40							

Flexowriter Printouts (cont.)

	IA	FP40							
40	24	52300	13167	A	P	E	△	F	U
41	46	46220	15267	L	L	.	△	P	U
42	66	01503	07101	T	△	N	E	W	△
43	04	10030	30131	I	5	0	0	△	F
44	51	51660	16624	O	0	T	△	T	A
45	52	30015	15001	P	E	△	O	N	△
46	65	30547	05177	S	E	R	V	O	77
47	0	0	0	[					]
50	65	66245	46601	S	T	A	R	T	△
51	66	51012	65150	T	0	△	C	O	N
52	66	34506	73001	T	I	N	U	E	△
53	46	34656	63450	L	I	S	T	I	N
54	32	22010	10101	G	.	△	△	△	△
55	01	07220	10101	△	4	.	△	△	△
56	01	12220	10101	△	7	.	△	△	△
	CA	FP57							

} Optional list-  
ing tape num-  
bers for 5 or  
7 servo layout

	IA	FC	
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	0	0	10
5	0	0	4
6	0	24	24
7	0	0	15
10	0	0	2
11	0	0	3
12	0	0	77
13	0	0	7777
14	0	7	77777
15	0	0	67
16	0	0	6667
17	0	0	22
20	06	0	0
21	0	0	5
22	04	0	0
23	0	77777	0
24	0	2	0
25	01	0	0
26	02	0	0
27	0	0	6
30	0	0	30
31	0	0	171
32	0	0	77777
33	0	7	0
34	76	0	0
35	0	0	14
36	0	0	36
37	0	4	0
40	0	3	0
41	0	77770	0
42	0	0	167
43	0	10	0
44	0	0	16
45	0	0	11
46	30	30200	0
47	01	01010	10000
50	0	7700	0
51	0	07777	0
52	0	166	10
53	0	167	1
54	0	777	0
55	0	170	0

Zero  
1 in "v"  
1 in "u"  
1 in "u" & "v"

20₁₀

Mask 1st XS3 digit  
Mask 1st & 2nd XS3 digits  
Mask 1st, 2nd & 3rd XS3 digits

"u" mask

XS3 space

# const. allowed in 1st blk.  
Const. Pool on obj. prog. tape (u).  
Minimum # blks. preceding 1st  
seg. lab. blk. on obj. prog. tape (%)

56	0	0	170	
57	0	0	30000	
60	0	23000	0	
61	0	25000	0	
62	0	26000	0	
63	0	30000	0	
64	0	50000	0	
65	0	60000	0	
66	0	1100	0	Max. # blks. input buffer (9 ₁₀ or 11 ₈ )
67	0	0	70	Limit for line count when new section next (56 ₁₀ )
70	0	0	100	Limit for line count when same section next (64 ₁₀ )
71	0	1200	0	Max. # blks. + 1 (10 ₁₀ or 12 ₈ ) in input buffer
72	0	100	0	
73	0	0	BL	Listing tape block limit (1200 ₁₀ per Univac sys. convention)
74	03	0	0	
75	0	77	0	
76	0	0	7	
77	0	10000	0	
100	0	0	12	
101	0	0	303	
	CA	FC102		

## Relative Constants

	IA	RC		
0	0	IB	0	Init. add. input buff.
1	0	OB	OB	Init. add. output buff.
2	TP	IB2070	CW4	Last add. input buffer + 1
3	0	OB264	OB264	
4	0	XS34	HC36	Add. stored subs. var. col. hdg.; ent. add. sent. hdgs W/cont.
5	0	XS40	HV7	Add. stored subs. var. underscores; ent. add. subs. var. drum W/cont.
6	0	0	OD4	To preset (A) → (A2)
7	0	0	BD	To preset (B) → (B2)
10	0	XS55	HC45	Add. stored non-subs. var. col. hdg.;ent. add. term. hdgs W/cont.
11	0	XS60	HV11	Add. stored non-subs. var. under- scores;ent. add. subs. var. (core) hdgs. W/cont.
12	0	LB170	0	Init. add. XS3 Sym. list
13	0	IB1	0	
14	AT	CW1	CW1	
15	0	NL	0	Base add. statements in sent. no. list
16	0	NL1000	0	Base add. subs. var. EQ. in sent. no. list
17	0	NL2000	0	Base add. non-subs. var. EQ. in sent. no. list
20	0	NL3000	0	Base add. pseudo Ops. in sent. no. list
21	40	NL3100	0	Base add. lib. rtns. in sent. no. list (Ind. bit in Op. code)
22	0	DL167	HC55	Add. 1st const. - 1 in input buffer
23	TP	FB170	Q	
24	0	FD	NP15	Init. add. Op. file IV on drum; Preset one shot jump page no. rtn.
25	0	FL	NP22	Init. add. Op. file IV in core; Preset one shot jump page no. rtn.
26	0	0	SB	Init. add. statement buffer
27	TP	RF	FD	Init. add. Op. file IV (drum) buffer in "v"
30	0	0	RB	Init add routine buffer
31	TJ	RC2	ED14	
32	MJ	0	ED14	
33	0	DL	HC5	
34	0	TB	HC27	
35	TP	RB	RF	Init. add. routine file (drum) in "v"
36	0	0	SB170	Limit value for statement buff. (last add. + 1)
37	0	0	RB170	Limit value for routine buff. (last add. + 1)

40	TP	FL170	CT	Limit value file list (last add. + 1 in "u")
41	0	OB740	OB740	Limit value output buff. (last add. + 1)
42	0	LB	DS4	Init. add. list buffer
43	07	0	0	
	CA	RC44		

Explanation of Counters, Indexes, Temps., Etc. (CT)

CT0	[0	0	0]	Temp. 1 curr. subs. var. CW. running add. etc.
1	0	0	[0]	Index C ₁
2	0	0	[0]	Index C ₂
3	0	0	[0]	Index C ₃
4	0	0	[0]	Index C ₄
5	0	0	[0]	Index C ₅
6	0	[0]	[0]	Output buff. add. (next avail. blkt.)
7	0	0	[0]	Line count (next avail. line)
10	0	0	0	1st page no. word
11	0	0	0	2nd page no. word
12	0	0	0	1st segment no. word
13	0	0	0	2nd segment no. word
14	0	0	0	Count of blocks on listing tape
15	[0	0	0]	Temp. 2
16	0	0	[0]	Seg. no. (octal)
17	0	0	[0]	# blks. in Term.
20	0[0	000] 00	0	# full blks. seg. + Pref.
21	0	[0]	[0]	# lines part. blk. and H.S.S.
22	RJ	[0]	[0]	Pref. exit and entry in "u" & "v"
23	0	0	[0]	# lines preface

} 3rd-  
8th  
Lines  
seg.  
lab.  
blk.